



BEST AVAILABLE COPY

JP5241741

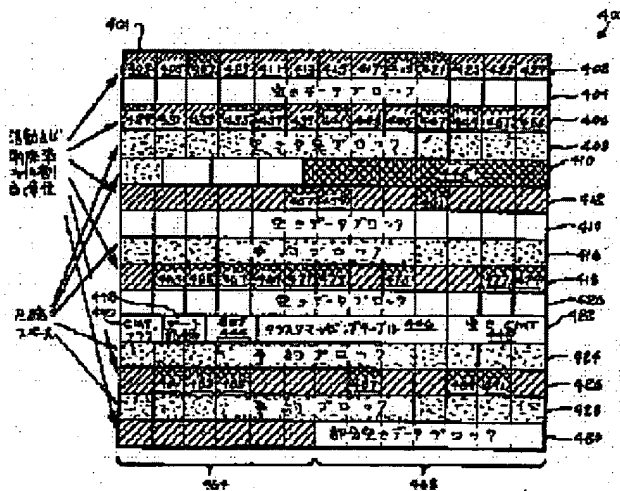
**Patent number:** JP5241741  
**Publication date:** 1993-09-21  
**Inventor:** JIERARUDO ESU HORUTSUHAMAA; KAATO  
BURAIA ROBINSON  
**Applicant:** INTEL CORP  
**Classification:**  
- international: G06F3/08; G11C16/06; G06F12/00  
- european: G06F3/06E; G06F12/02D2E2  
**Application number:** JP19910354041 19911219  
**Priority number(s):** US19900635988 19901231

Also published as:

 GB2251323 (A)  
 DE4143072 (A1)

## Abstract of JP5241741

**PURPOSE:** To execute disk emulation about a non-volatile semiconductor memory to be erased in a block unit. **CONSTITUTION:** The non-volatile semiconductor memory to be erased in a block unit is explained. The non-volatile semiconductor memory contains an active block to store first data and reserved blocks 416, 424, 428 to store second data. The second data is the copy of the first data. That copy is generated during clean-up operation prior to the erasure of the active block. The non-volatile semiconductor memory contains further a mapping table to map the logical address of an assignment unit on the physical address of one sector in the non-volatile semiconductor memory.



Data supplied from the esp@cenet database - Worldwide

(11)特許出願公開番号

特開平5-241741

(43)公開日 平成5年(1993)9月21日

(51)Int.Cl. <sup>5</sup>	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 3/08	H	7165-5B		
G 1 1 C 16/06				
// G 0 6 F 12/00	5 0 1 H	7232-5B		
		9191-5L	G 1 1 C 17/ 00	3 0 9 C

審査請求 未請求 請求項の数 2 (全 28 頁)

(21)出願番号 特願平3-354041

(22)出願日 平成3年(1991)12月19日

(31)優先権主張番号 635,988

(32)優先日 1990年12月31日

(33)優先權主張国 米国 (U S)

(71)出願人 591003943

インテル・コーポレーション

アメリカ合衆国 95052 カリフォルニア  
州・サンタクララ・ミッション カレッジ  
ブーレバード・2200

(72)発明者 ジェラルド・エス・ホルツハマー

アメリカ合衆国 97007 オレゴン州・ア  
ロハ・サウスウェスト イーグル クレス  
ト テラス・8520

(72)発明者 カート・ブライアン・ロビンソン

アメリカ合衆国 95658 カリフォルニア  
州・ニューキャスル・ネイバス レイン・  
2216

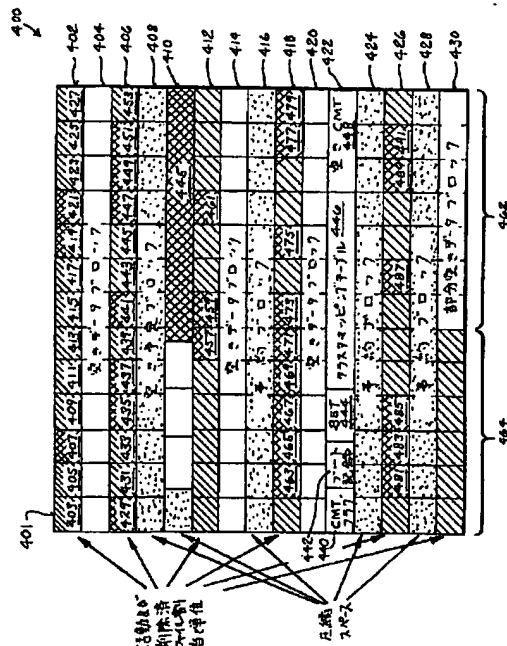
(74)代理人 弁理士 山川 政樹

(54)【発明の名称】 不揮発性半導体メモリ及びこれを使用したコンピュータシステム

(57) 【要約】

【目的】 ブロック単位で消去される不揮発性半導体メモリについてディスクエミュレーションを実行すること。

【構成】 ブロック単位で消去される不揮発性半導体メモリを説明する。不揮発性半導体メモリは第1のデータを記憶する活動ブロックと、第2のデータを記憶する予約ブロックとを含む。第2のデータは第1のデータのコピーである。そのコピーは、活動ブロックの消去に先立ってクリーンアップ動作の間に作成される。不揮発性半導体メモリは、割当て単位の論理アドレスを不揮発性半導体メモリ内部の1つのセクターの物理アドレスにマッピングするマッピングテーブルをさらに含む。



## 【特許請求の範囲】

【請求項1】 ブロック単位で消去される不揮発性半導体メモリにおいて、

(A) 第1のデータを記憶する活動ブロックと；

(B) 活動ブロックの消去に先立ってクリーンアップ動作の間に作成される第1のデータのコピーである第2のデータを記憶する予約ブロックと；

(C) 割当て装置の論理アドレスを不揮発性半導体メモリ内部の1つのセクターの物理アドレスにマッピングするマッピングテーブルと；を具備する不揮発性半導体メモリ。

【請求項2】 (A) 中央処理装置と；

(B) (1) 第1のデータを記憶する活動ブロックと；

(2) 活動ブロックの消去に先立ってクリーンアップ動作の間に作成される第1のデータのコピーである第2のデータを記憶する予約ブロックと；(3) 割当て装置の論理アドレスを不揮発性半導体内部の1つのセクターの物理アドレスにマッピングするマッピングテーブルとを具備するブロック単位で消去可能であり、中央処理装置によりアクセスされる不揮発性半導体メモリと；

(C) 不揮発性半導体メモリを制御するためのコードを記憶する記憶手段と；を具備するコンピュータシステム。

## 【発明の詳細な説明】

## 【0001】

【産業上の利用分野】本発明はコンピュータ記憶システムのアーキテクチャの分野に関し、特に、大形ブロック単位で消去可能な不揮発性半導体メモリにおけるディスクエミュレーションに関する。

## 【0002】

【従来の技術】ある種の従来のパーソナルコンピュータシステムは、数種類の記憶システム、すなわち、読取り専用メモリ(「ROM」)、ランダムアクセスメモリ(「RAM」)、大容量記憶のためのハード(すなわち、固定)ディスクドライブ及び着脱自在磁気フロッピーディスクへの記憶のためのフロッピーディスクドライブ等に結合するマイクロプロセッサ(中央処理装置ともいう)を含む。また、フロッピーディスクをディスクともいう。このような従来のパーソナルコンピュータシステムは、通常、従来のパーソナルコンピュータシステムそれぞれの一部を形成する記憶システムに特に適応するアーキテクチャを有する。

【0003】ROMに記憶されているROMモニターと呼ばれるプログラムと組み合わせたROMをファームウェアと呼ぶ。ROM基本入出力システム(「BIOS」)モジュールは、ROMに記憶され、ある種のパーソナルコンピュータではオペレーティングシステムにより使用されるROMモニターの一種である。ROM BIOSモジュールは、通常、(1)キーボード、ディスクドライブ及びプリンタを含むいくつかのハードウェア

を駆動するドライバと、(2)パワーオン自己試験プログラム(「POST」)と、(3)システムを初期設定する始動ルーチンと、(4)ディスク又はハードディスクからブート、すなわち、第1のセクターを読取るロードプログラムとを含む。

【0004】パーソナルコンピュータがオンした後、BIOSモジュールのPOSTプログラムが実行され、BIOS始動ルーチンはいくつかの初期設定動作を実行し、ロードプログラムはディスク又はハードディスクからブートセクターの内容を読取る。そのブートセクターはコンピュータのオペレーティングシステムのロードプログラムを含む。オペレーティングシステムのそのロードプログラムはオペレーティングシステムの一部をディスク又はハードディスクからRAMにロードする。

【0005】パーソナルコンピュータのオペレーティングシステムは指令を処理し、プログラムの実行を制御し、コンピュータシステムのハードウェア資源及びソフトウェア資源を監督する。従来のオペレーティングシステムの一種として、ワシントン州レッドモンドのMicrosoft Corporationから販売されているMS-DOSがある。MS-DOSオペレーティングシステムは先に述べたロードプログラムと、MS-DOS BIOSと、MS-DOS核と、ユーザーインタフェースと、ユーティリティプログラムとを含む。

【0006】MS-DOS BIOSは(1)ROM BIOSのドライバの構成要素を拡張し且つ使用する新たなドライバと、(2)MS-DOS BIOSドライバの初期設定ルーチンと、(3)別のロードプログラムとを含む。MS-DOS BIOSの新たなドライバを、BIOS拡張ソフトウェア又はBIOS ROM拡張ソフトウェアともいう。従来のパーソナルコンピュータによっては、ROMの32キロバイト又は64キロバイトのセクションに記憶されているソフトウェアと、それより小さい(通常は2キロバイト)BIOS ROM拡張ソフトウェアとにより全ての入出力機能を駆動するものがある。MS-DOS BIOSドライバの初期設定ルーチンは著作権情報を表示すると共に、新たなドライバに関する割り込みテーブルを調整する。MS-DOS BIOSのロードプログラムはオペレーティングシステムの残り部分をロードする。

【0007】MS-DOS核はBIOSと、アプリケーションプログラムとの間のシェルである。MS-DOS核はアプリケーションプログラムの実行を開始し、アプリケーションプログラムについてメモリを割当て、アプリケーションプログラムとハードウェアとの間にアプリケーションプログラムインタフェースを形成し、ファイルの読取り及び書き込みを管理する。

【0008】MS-DOSのユーザーインタフェースはユーザーに情報を提供する。ユーザーインタフェース

は、ユーザーに指令の入力を促すプロンプトを提供する。オペレーティングシステムが制御状態にあるとき、ユーザーインタフェースはシステムのマネージャとして動作する。MS-DOSのユーティリティプログラムはMS-DOSに関わるいくつかの有用な機能を実行する。それらの機能は(1)のディスクット又はハードディスクのフォーマット作成と、(2)ディスク又はハードディスクの検査とを含む。

【0009】上述のようなパーソナルコンピュータシステムと共に使用するハードディスクや、ディスクは不揮発性記憶システムである。すなわち、コンピュータへの給電が停止したとき、データは失われない。また、ハードディスクとディスクはブロック記憶装置に属しており、従って、データはそれらの記憶装置との間でブロック単位で転送される。データはハードディスクや、ディスクの同心トラックに物理的に記憶される。各トラックは複数のセクターから構成されている。1つのセクターは、通常、512バートの固定長さを有する。通常、セクター記述項への書込み及びセクター記述項からの読取りを常に実行するのは、パーソナルコンピュータシステムのディスクコントローラと、物理装置ドライバである。

【0010】MS-DOSの場合、クラスタは論理的にアドレッシング可能な最小の記憶単位である。ハードディスクの中には、クラスタごとに4つのセクターを含むものがある。また、他のハードディスクや、高密度3.5インチディスクにおいては、1つのクラスタは単一のセクターである。ハードディスクの各区分領域は、独自のオペレーティングシステムを含むことができる論理サブシステムを形成する。フォーマット作成済ハードディスクの第1のセクターにある区分テーブルは、区分に関する情報を含む。ハードディスクとディスクは、最初に使用する前にフォーマット作成される。低レベルのフォーマット作成は各トラックを複数のセクターに分割し、トラックに沿って均一に配分された位置に識別(「ID」)セクターヘッダを配置する。高レベルフォーマット作成はクラスタを確定し、いくつかのディスク領域を初期設定し、ディスクのデータ受信準備を整える。

【0011】装置ドライバ及びBIOSレベルでは、ディスク要求は、ドライブ、ヘッド、シリンダ/トラック、セクター及び長さを指示する「チュープル」(tuple)により表される。DOSレベルと、BIOSレベルにおいて論理セクター数はわかっている。DOSはクラスタエンティティ単位でディスクからの読取り及びディスクへの書込みを実行する。

【0012】図1は、従来のMS-DOSオペレーティングシステムの論理的編成を示す。MS-DOSオペレーティングシステムの場合、ディスク2はシステム領域4と、データ領域9の2つの論理領域に分割される。シ

ステム領域はブート記録3と、ファイル割当てテーブル(「FAT」)5と、ルートディレクトリ記述項を含むルートディレクトリ領域7とを含む。データ領域9は、アプリケーションプログラムと、データと、サブディレクトリ情報とを記憶するために使用されるファイルを含む。ブート記録3は、オペレーティングシステムをロードするブートストラップロードプログラムを含む。ブート記録3は、フォーマット作成DOSのASCII名と、ディスクのセクターごとのバイト数と、クラスタごとのセクター数と、ブート記録中のセクター数と、ファイル割当てテーブル記述項(又はクラスタ)のコピー数と、ルートディレクトリ記述項の数と、区分ごとのセクター数と、ディスクの型番号と、ファイル割当てテーブルごとのセクター数と、トラックごとのセクター数と、ディスクごとの面の数と、予約セクター、すなわち、かげのセクターの数と、物理ドライブ番号と、拡張ブートセクタリングナチュアと、ボリューム識別と、ボリュームテーブルとに関する情報をさらに含む。

【0013】ルートディレクトリ7はそれぞれがファイルのいくつかの属性を記述する複数の32ビット記述項から成るテーブルである。通常、ルートディレクトリ7を構成している各ディレクトリ記述項はファイル名と、ファイル拡張と、属性フラグと、ファイルの日時スタンプと、ファイルを構成するクラスタに関する開始クラスタ番号と、ファイルサイズとを含む。ディスクの各ファイルは1つ又は複数のクラスタから構成されている。ファイル割当てテーブル5は、ファイルを構成しているクラスタが互いにどのように連係するかという連鎖の形態をとる記録を含む。典型的なFAT5は、クラスタごとに1つずつ、2バイト記述項のリストを含む。従来のFATの中には、FAT記述項が2バイトを超える長さのものもある。各FAT記述項の長さはクラスタの総数によって決まる。1つのファイルのディレクトリ記述項はそのファイルの開始クラスタ番号を含み、オペレーティングシステムはその開始クラスタ番号を使用して、ファイル割当てテーブルをアクセスする。各FAT記述項は、ファイルの次のクラスタに対するポインタである。従って、その第1回のアクセスにより検索されるFAT記述項は、ファイルを構成している次のクラスタのクラスタ番号を含む。オペレーティングシステムはその次のクラスタ番号を使用して、さらに別のクラスタ番号を検索するためにFATをアクセスし、このプロセスをFAT5の中の特別のマーカに達するまで続ける。

【0014】ディスクにおけるファイル構造は樹木形である。ルートディレクトリの記述項はサブディレクトリに対するポインタとなることができる。サブディレクトリは入れ子形構成をとっていても良い。従来のある種のパーソナルコンピュータの場合、ハードディスク及びディスクの使用と関連して、ハードディスクドライブや、フロッピーディスクドライブが多数の機械的構要素

10

20

30

40

50

素を含む、物理的にかなり大形の装置であるという欠点が見られる。その大きさは、パーソナルコンピュータの他の多くの部品を構成している集積回路の小ささとは対照的である。さらに、従来の典型的なハードディスクドライブ及びフロッピーディスクドライブが相対的に大形であることは、ポータブルパーソナルコンピュータをさらに小形にし、その可搬性を向上させる上での障害となっている。

【0015】従来のハードディスクドライブ及びフロッピーディスクドライブと関連するもう1つの欠点は、それらの装置がパーソナルコンピュータの他の部品を構成している集積回路と比較して相対的に大量の電力を消費するという点である。従来のハードディスクドライブ及びフロッピーディスクドライブと関連するさらに別の欠点は、過剰な衝撃や震動を受けた場合又はちり又は他の大気中汚染物質にさらされた場合に故障しやすいことである。

【0016】従来の別の種類の不揮発性コンピュータメモリは、フラッシュ電気的消去可能プログラム可能読み専用メモリ（「フラッシュEEPROM」）である。フラッシュEEPROMはユーザーによりプログラム可能であり、一旦プログラムしてしまえば、消去されるまで、EEPROMはデータを保持する。フラッシュEEPROMの電気的消去は、装置のメモリの全ての内容を1回の比較的短時間の動作で消去する。その後、新たなコードによってフラッシュEEPROMをプログラムすれば良い。ところが、従来のフラッシュEEPROMの1つの型には、あらかじめ消去せずに個々のビットセルを論理値0から論理値1に重ね書きすることが不可能であるという欠点がある。従来のフラッシュEEPROMの1つの型のもう1つの欠点は、大きなブロック単位で又は装置全体を消去するように消去、すなわち、論理値1の状態にリセットしなければならないことである。

【0017】従来のフラッシュEEPROMの1つの型のさらに別の欠点は、フラッシュEEPROMが故障に至るまでのフラッシュEEPROMの消去サイクル及び書き込みサイクルの数に限りがあることである。ある種の従来のフラッシュEEPROMと関連する重ね書き及び消去に関する制限は、場合によっては、パーソナルコンピュータシステムにおけるフラッシュEEPROMの有用性を限定してしまっていた。

【0018】

【発明が解決しようとする課題】本発明の目的の1つは、ブロック単位で消去される不揮発性半導体メモリについてディスクエミュレーションを実行することである。本発明の別の目的は、構造の安全性及び信頼性を最大限に高めるのを助ける予約ブロックを含むブロック単位で消去される不揮発性半導体メモリについてディスクエミュレーションを実行することである。本発明の別の

目的は、ブロック単位で消去可能であり且つディスクエミュレーションを含む不揮発性半導体メモリを含むコンピュータシステムを提供することである。

【0019】

【課題を解決するための手段】ブロック単位で消去される不揮発性半導体メモリを説明する。不揮発性半導体メモリは第1のデータを記憶する活動ブロックと、第2のデータを記憶する予約ブロックとを含む。第2のデータは第1のデータのコピーである。そのコピーは、活動ブロックの消去に先立ってクリーンアップ動作の間に作成される。不揮発性半導体メモリは、割当て単位の論理アドレスを不揮発性半導体メモリ内部の1つのセクターの物理アドレスにマッピングするマッピングテーブルをさらに含む。本発明のその他の目的、特徴及び利点は、添付の図面及び以下の詳細な説明から明白になるであろう。本発明を添付の図面の図に例示してあるが、それらは限定的な意味をもつものではない。尚、図面中、同じ図中符号は同様の要素を指す。

【0020】

【実施例】図2は、好ましいファイルシステム構造32の1つを備えたパーソナルコンピュータシステム10を示す。ファイルシステム32は1つのフラッシュEEPROM又はフラッシュメモリアレイ34を形成する複数のフラッシュEEPROMを含む。フラッシュEEPROMは半導体メモリ的一种である。ファイルシステム32は、システムRAM22に記憶されているソフトウェアファイルシステムドライバ28をさらに含む。パーソナルコンピュータシステム10は中央処理装置（「CPU」）12と、バス18とをさらに含む。以下にさらに詳細に説明するが、フラッシュメモリアレイ34は固定ディスクドライブ又はフロッピーディスクドライブのいずれかをエミュレートするような構造である。

【0021】システムRAM22は、アプリケーションプログラム24を記憶するためのスペースを有する。本発明の一実施例では、システムRAM22は1メガバイトの大きさである。システムRAM22は、他のアプリケーションプログラムと、データとを記憶するためのスペースをさらに含む。本発明の一実施例においては、システムRAM22はコンピュータシステム10に関わるオペレーティングシステム26を1つ記憶している。別の実施例では、システムRAM22はさらに2つ以上のオペレーティングシステムを記憶している。

【0022】システムRAM22はファイルシステムドライバソフトウェア28をさらに含む。ファイルシステムドライバソフトウェア28はフラッシュメモリアレイ34のファイル構造を規定する。パーソナルコンピュータシステム10では、中央処理装置12はフラッシュメモリアレイ34の全てのファイルシステム管理ユーティリティを処理するためにファイルシステムドライバソフトウェア28のフォアグラウンドタスクソフトウェア

ーチンを実行する。ファイルシステムドライバ28と、フラッシュメモリアレイのファイル構造については以下にさらに詳細に説明する。

【0023】システムRAM22がアプリケーションプログラム24、オペレーティングシステム26及びファイルシステムドライバ28を記憶するばかりではなく、コンピュータシステム10がオフ又はオンのいずれかである時間中、アプリケーションプログラム24、オペレーティングシステム26及びファイルシステムドライバ28はフラッシュメモリアレイ34にも記憶される。フラッシュメモリアレイ34は2つ以上のアプリケーションプログラムと、2つ以上のオペレーティングシステムと、2つ以上のファイルシステムドライバとを記憶することができる。フラッシュメモリアレイ34はデータファイルと、ディレクトリ情報とをさらに記憶する。ウォームブート、コールドブートのいずれかに対して、ファイルシステムドライバソフトウェア28をフラッシュメモリアレイ34からシステムRAM22にロードする。要するに、フラッシュメモリアレイはハードディスクの代わりとなる大容量記憶装置として動作するのである。

【0024】コンピュータシステム10は、コンピュータシステム10のBIOSソフトウェアを記憶するROM14をさらに含む。本発明の一実施例では、ROM14に記憶されるBIOSは64キロバイトのBIOSと、それに追加されるそれぞれ2キロバイトのBIOS

ROM拡張部分の双方を含む。ファイルシステムドライバソフトウェア28は、実行に先立って、ROM14ではなくフラッシュメモリアレイ34に記憶される。コンピュータシステム10のパワーアップ後にROM14のBIOSプログラムが実行されるとき、ファイルシステムドライバ28とオペレーティングシステム26をフラッシュメモリアレイ34からバス18を介してシステムRAM22にロードする。そこで、コンピュータシステム10のユーザーはオペレーティングシステム26を使用して、アプリケーションプログラム24をシステムRAM22にロードする。さらに、ユーザーはデータ、他のアプリケーションプログラム及び他のオペレーティングシステムをシステムRAM22にロードすることもできる。

【0025】本発明の一実施例においては、フラッシュメモリアレイ34は、それぞれが多数のビットから成るブロックの形態で消去される。別の実施例では、フラッシュメモリアレイ34をことごとく消去する。フラッシュメモリは、消去に関して、従来の電氣的消去可能プログラム可能読取り専用メモリ(「EEPROM」)とは異なる。従来のEEPROMはバイト消去を個々に制御するためにセレクトトランジスタを使用する。これに対し、フラッシュメモリは1つ1つのトランジスタセルによってはるかに高い密度を得ている。消去モードの間、1つのブロック又はチップ全体にあるあらゆるメモリセ

ルのソースに同時に高電圧を供給する。その結果、アレイ全体又はブロック全体が消去されることになる。

【0026】フラッシュメモリアレイ34の場合、論理値「1」は、1つのビットセルと関連するフローティングゲートに蓄積されている電子がもしあるとしてもわずかなであるということを表わす。論理値「0」は、ビットセルと関連するフローティングゲートに多数の電子に蓄積されていることを表わす。フラッシュメモリ34の消去後、フラッシュメモリアレイ34の各ビットセルに論理値1が記憶される。あらかじめ消去することなく、フラッシュメモリアレイ34の個々のビットセルを論理値0から論理値1に重ね書きするのは不可能である。ただし、消去状態と関連する本来存在している数の電子を含むフローティングゲートに単に電子を追加するという結果を生じるのであれば、フラッシュメモリアレイ34の個々のビットセルを論理値1から論理値0に重ね書きすることはできる。

【0027】フラッシュメモリアレイ34は、(1)一度に1ビットずつ、(2)一度に1語ずつ、(3)一度に一群の語ずつという3つの方法のいずれか1つでプログラム可能である。1つの語は1つのメモリシステムアドレス又は1つの装置と関連する複数のメモリビットから構成される。一度にプログラムされる語群は、フラッシュメモリアレイ34の1つの消去ブロックと同じ大きさであることが可能である。フラッシュメモリに関するプログラム動作を書込み動作ともいう。先に述べた通り、あらかじめ消去することなく、フラッシュメモリアレイ34の各ビットを論理値0の状態から論理値1の状態に重ね書きするのは不可能である。論理値0から論理値1に重ね書きするのに先立って、このように消去の必要があることにより、フラッシュメモリと関連する機能動作が導入される。

【0028】フラッシュメモリアレイ34は実行可能コードと、非実行データの双方を記憶する。以下の詳細な説明中、「データ」という総称は、(1)非実行データのみ、又は(2)実行可能コードと非実行データの双方のいずれかを表すために使用される。フラッシュメモリアレイ34と関連する読取り動作は、他の読取り専用メモリ装置と関連する読取り動作とごく似ている。一実施例では、フラッシュメモリアレイ34の読取り動作は135ナノ秒を要する。これに対し、フラッシュメモリアレイ34の書込み動作と消去動作はそれより著しく遅い。本発明の一実施例においては、フラッシュメモリアレイ34の1ブロックを消去する時間は1秒である。フラッシュメモリアレイ34に1つの語を書込む時間は10ミリ秒である。従って、フラッシュメモリアレイ34と関連する読取り動作と、書込み動作及び消去時間は本質的に非対称である。すなわち、書込み動作と消去動作は読取り動作より著しく遅いのである。

【0029】さらに、フラッシュメモリアレイ34の各

フラッシュメモリは、消去+書込み（すなわち、プログラム）動作に関して限られた耐久時間を有する。たとえば、一実施例では、フラッシュメモリアレイ34が故障し始めるか、又は消去/書込み性能の劣化を示す前に、少なくとも10000回の消去/プログラムサイクルに耐えることができる。従って、本発明の一実施例においては、フラッシュメモリアレイ34が遂行できる消去/書込みサイクルの数に関して限界がある。一方、フラッシュメモリアレイ34からの読取りが可能なサイクルの数に関しては、そのような最長寿命というようなものは存在していない。

【0030】本発明の一実施例では、フラッシュメモリアレイ34を構成している各フラッシュメモリデバイスは、それぞれのメモリデバイスの中に複数の消去ブロックを有する。その実施例の場合、隣接するブロックの消去/プログラミングサイクリングには制限がある。隣接ブロックとは、共通の行結合又は共通の列結合を共用するブロックである。隣接ブロックのサイクリングの制限は、電気的データ妨害状態を阻止するためになされる。サイクリングがより頻繁に起こるブロックをホットブロックという。サイクリングが起こらないか、さほど頻繁にサイクリングしないブロックをコールドブロックという。重要なパラメータは、隣接ブロックとそれぞれ関連するサイクルの数の差出ある。ホットブロックは、フラッシュメモリアレイ34のコールドブロックではデータが不利な影響を受けるようなサイクルカウントに最終的に達することができる。このホットブロック/コールドブロックインタラクションを回避する方法の1つは、フラッシュメモリアレイ34のホットブロックにサイクルの制限を課すことである。ホットブロック/コールドブロックインタラクションの不利な影響を最小限に抑える別の方法は、フラッシュメモリアレイ34のコールドブロックを周期的に消去し、再プログラムすることである。このようなコールドブロックの周期的な消去と再プログラミングを、コールドブロックの再生という。以下にさらに詳細に説明するように、本発明の好ましいファイル構造は、フラッシュメモリアレイ34を構成するフラッシュEEPROMの上述のような機能上の特徴を考慮に入れている。

【0031】図3は、好ましいファイルシステム構造58の1つを備えたパーソナルコンピュータシステム40を示す。パーソナルコンピュータシステム40は中央処理装置12と、ROM42とを含む。ROM42はBIOSソフトウェア43と、BIOS拡張ソフトウェア44とを記憶している。パーソナルコンピュータシステム40はバス48と、システムRAM52とをさらに含む。システムRAM52はアプリケーションプログラム24と、オペレーティングシステム26とを記憶している。パーソナルコンピュータシステム40はRAMバッファ62と、フラッシュメモリアレイ64とをさらに含

む。RAMバッファ62は、フラッシュメモリアレイ64と、システムRAM52との間でのファイルの転送について緩衝を実行する。RAMバッファ62はフラッシュメモリアレイ64に記憶されているファイルのディレクトリの構成又はその更新についても緩衝を実行する。1つ又は2つ以上のフラッシュEEPROMがフラッシュメモリアレイ64を構成している。フラッシュメモリアレイ64はアプリケーションプログラム24と、オペレーティングシステム26と、データと、他のアプリケーションプログラムと、他のオペレーティングシステムとを記憶する。フラッシュメモリアレイ64からバス48を介してシステムRAM52へ、オペレーティングシステム26、アプリケーションプログラム24、他のプログラム及びデータを転送することができる。ファイルシステム構造58はフラッシュメモリアレイ64と、RAMバッファ62と、ROM BIOS拡張ソフトウェア44と、ROM BIOSソフトウェア43の一部とを含む。

【0032】フラッシュメモリアレイ64のファイル構造を規定するファイルシステムドライバソフトウェアは、ROM42のROM BIOSソフトウェア43及びROM BIOS拡張ソフトウェア44の一部を形成する。図3に示す実施例では、ファイルシステムドライバはパーソナルコンピュータシステム40のファームウェアの一部であり、それと一体である。BIOSソフトウェア43とBIOS拡張ソフトウェア44は、直接マッピングされるか又はページングされるメモリブロックである。従って、図3に示す実施例においては、ファイルシステムドライバソフトウェアはシステムRAM52、フラッシュメモリアレイ64のいずれにも記憶されない。図3に示す実施例の場合、ROM BIOS拡張ソフトウェア44の一部として、BIOS割込み13H拡張ソフトウェアが設けられている。この割込み13H拡張ソフトウェアは、初期設定中、標準割込み13Hベクトルにパッチされる。フラッシュメモリアレイ64について目標とならないBIOS要求は、旧BIOS割込み13Hハンドラへ転送される。

【0033】図4は、別の好ましいファイルシステム構造94を備えたパーソナルコンピュータシステム70を示す。パーソナルコンピュータシステム70は中央処理装置12と、システムRAM72とを含む。1つの好ましい実施例では、システムRAM72はアプリケーションプログラム24と、オペレーティングシステム26とを含む。パーソナルコンピュータシステム70はバス88と、ROM78とをさらに含む。ROM78はBIOSソフトウェア79と、BIOS拡張ソフトウェア80とを記憶している。パーソナルコンピュータシステム70はコントローラ92と、フラッシュメモリアレイ96とをさらに含む。フラッシュメモリアレイ96は複数のフラッシュEEPROMから構成されている。フラッシュ

11

メモリアレイ96はアプリケーションプログラム24と、オペレーティングシステム26と、データと、他のアプリケーションプログラムと、他のオペレーティングシステムとを記憶するが、バス88を介してこれらをシステムRAM72にロードすることができる。ファイルシステム構造94はコントローラ92と、フラッシュメモリアレイ96と、ROM BIOS拡張ソフトウェア80と、ROM BIOSソフトウェア79の一部とから構成される。

【0034】図4に示すパーソナルコンピュータシステム70は、フラッシュメモリアレイ96に対して、完全ハードウェア駆動ファイル構造を有する。コントローラ92は、フラッシュメモリアレイ96に関わるファイル構造を規定し且つ制御するための特別な専用コントローラである。コントローラ92は、監督制御のためのマイクロコントローラ100を含む。コントローラ92は制御論理106をさらに含む。本発明の好ましい一実施例では、制御論理106はプログラム可能論理アレイ（「PLA」）から構成されている。制御論理106は、コントローラ92の動作を制御するデジタル論理を含む。

【0035】一実施例においては、コントローラ92は、ファイルシステムドライバソフトウェアを記憶するROM98を含む。ファイルシステムドライバソフトウェアはマイクロコントローラ100により実行可能であり、フラッシュメモリアレイ96のファイル構造を規定する。別の実施例では、実行可能であり且つフラッシュメモリアレイ96のファイル構造を規定するファイルシステムドライバソフトウェアはROM BIOSソフトウェア79及びROM BIOS拡張ソフトウェア80の一部を形成している。コントローラ92はバッファRAM102を含む。バッファRAM102は、ファイルの転送と、ディレクトリの作成及び更新とについて緩衝を実行する。コントローラ92はバッファ/マルチプレクサ104をさらに含む。バッファ/マルチプレクサ104はファイルの転送について緩衝を実行すると共に、フラッシュメモリアレイ96との間で送受信されるデータを多重化する。論理バッファ/マルチプレクサ回路104と関連する論理はどのような標準システムバスインタフェースにも適合することができる。たとえば、バッファ/マルチプレクサ104と関連する論理は、PCXT、PCAT（すなわち、IDE-工業規格アーキテクチャ）、EISA（すなわち、拡張工業規格アーキテクチャ）、MCA（すなわち、マイクロチャネルアーキテクチャ）、VME（すなわち、仮想機械環境）及びマルチバスなどの規格の1つ又は2つ以上に適合できる。

【0036】本発明の一実施例では、コントローラ92はパーソナルコンピュータシステム70の中に入っている。たとえば、パーソナルコンピュータシステム70の

12

システムボード（図示せず）にコントローラ92を配置することが可能であろう。別の実施例では、コントローラ92は、パーソナルコンピュータシステム70の外部又はパーソナルコンピュータシステム70の拡張スロット（図示せず）の内部にある入出力（「I/O」）装置であっても良い。この別の実施例の場合、バッファ/マルチプレクサ104と関連する論理はどのような標準I/Oインタフェースにも適合することができる。たとえば、バッファ/マルチプレクサ104はIDE、ST506、SCSI（すなわち、小型コンピュータシステムインタフェース）及びSA400（すなわち、フロッピーディスク規格）などのI/Oインタフェースの1つ又は2つ以上に適合できる。従って、この実施例については、パーソナルコンピュータシステム70はそれらの標準I/Oインタフェースの中の1つを介してコントローラ92と通信するものと考えられる。

【0037】図3のパーソナルコンピュータシステム40と、図4のパーソナルコンピュータシステム70は、それぞれ、対応するフラッシュメモリアレイのファイル構造を制御するのに専用のハードウェアを含む。パーソナルコンピュータシステム40の場合、その追加ハードウェアはRAMバッファ62である。パーソナルコンピュータシステム70の場合には、その追加ハードウェアはコントローラ92である。図3のRAMバッファ62と、図4のコントローラ92は、システムRAM52、フラッシュメモリアレイ64、システムRAM72又はフラッシュメモリアレイ96にファイルシステム制御コードをそれぞれ記憶する必要を回避する手段を構成する。RAMバッファ62と、コントローラ92はそれぞれデータを干渉する働きをするので、図2のシステムRAM52と、図4のシステムRAM72はファイルシステム構造58及び94のそれぞれについてデータバッファとして動作する必要はない。さらに、パーソナルコンピュータシステム40にデータバッファ62を含める、また、パーソナルコンピュータシステム70にコントローラ92を含めるということは、システムRAM52と、システムRAM72は、フラッシュメモリアレイ64及び96のファイル構造についてそれぞれスクラッチパッド領域として動作する必要がないことを表わしている。

【0038】図3のRAMバッファ62及び図4のコントローラ92が設けられているために、ファイルシステム構造58と、ファイルシステム構造96は、それぞれ、対応するパーソナルコンピュータシステムの基本システムハードウェアの一部を成すブート可能記憶領域を含むものとして認識されることになる。その結果、RAMバッファ62とコントローラ92は、それぞれ対応するパーソナルコンピュータシステムについてフラッシュメモリアレイのファイル構造の総合性能を向上させる働きをするのである。



【0039】図2～図4に示すパーソナルコンピュータシステム10、40及び70は構成が異なるので、コンピュータの特性も互いに異なる。図2のコンピュータシステム10では、ファイルシステムドライバ28は、フラッシュメモリアレイ34を管理するための全てのファイル管理ユーティリティを処理するフォアグラウンドタスクソフトウェアルーチンとして、中央処理装置12により実行される。ファイルシステムドライバ28はシステムRAM22の一部を占めるが、システムRAM22の大きさは限定されるであろう。たとえば、パーソナルコンピュータシステム10のオペレーティングシステム26がMS-DOSである場合、システムRAM22は1メガバイト分の「リアルモード」スペースに限定されると思われる。

【0040】図3のコンピュータシステム40に関していえば、フラッシュメモリアレイ64のファイル構造を規定するファイルシステムドライバソフトウェアは、ROM42に記憶されているROM BIOSソフトウェア43及びROM BIOS拡張ソフトウェア44の一部を形成する。ところが、ROM42における記憶スペースの量は限定されるであろう。たとえば、MS-DOS適合パーソナルコンピュータによっては、それぞれ2キロバイトのより小さなBIOS ROM拡張ソフトウェアを伴うROMの64キロバイトBIOS部分に記憶されているファームウェアによって、あらゆる入出力機能を駆動するものがある。ROM42のサイズが小さいと、ファイルシステムドライバソフトウェアはROM42で利用しうるスペースを越えてしまうであろう。

【0041】これに反して、コンピュータシステム70のハードウェアコントローラ92はファイルドライバを記憶しており、その結果、フラッシュメモリアレイ96、ROM78及びシステムRAM72に関してメモリに要求される条件は緩和される。さらに、ハードウェアコントローラ92はファイルシステム制御指令を簡略にする。たとえば、ハードウェアコントローラ92はファイルシステム制御指令を、他のいずれかのメモリスペースに存在しているコードを参照するラベルによりアクセス可能なソフトウェアである高レベル手順呼出しに変換する。本発明の一実施例では、簡略化されたファイルシステム制御指令、すなわち、ファイルドライバ制御指令はハードウェアコントローラ92のROM98に従来のBIOS型制御コードとして記憶される。これにより、固体ファイルシステム94を基本ブート可能装置として認識できるようになる。その結果、ファイルシステム呼出しは簡略化される。各ファイルシステム呼出しは、ファイルシステムドライバ（ファイルシステム制御コードともいう）を実行させる。

【0042】一実施例では、図3のコンピュータシステム70に対する各ファイルシステム呼出しは、ROM78に記憶されている主BIOSソフトウェア及び直接マ

ッピング形BIOS拡張ソフトウェアから命令を取り出すCPU12により直接実行される。別の実施例においては、各ファイルシステム呼出しは、ROM98に記憶されているBIOS拡張メモリブロックに存在するメモリページからCPU12により実行される。さらに別の実施例では、各ファイルシステム呼出しは、ハードウェアコントローラ92のROM98に記憶されているファイルシステムドライバからマイクロコントローラ100により実行される。好ましい実施例においては、ハードウェアコントローラ92は各ファイル関連タスクをホストCPU12からの単一の指令に縮小するが、これはホストCPU12のオフローディングを増加させるのに役立つ。コントローラ92は、入力データを局所RAMバッファスペース102に保持することにより、入力データをRAM形メモリ速度で受け入れる。ハードウェアコントローラ92はそれぞれ与えられるタスクを自動的に完了するが、一度に1つの動作を処理するのみである。

【0043】さらに、コントローラ92はホストCPU12に即時データアクセスに備えていづれのファイルシステムタスクをも中断させる。これは、緩衝しないとアクセス不可能なブロックに入ってしまうと思われる有効ファイル情報を緩衝する追加RAMを要求するとアクセス不可能ブロックは、書き込み動作又は消去動作に関連するブロックである。コントローラ92は中断に続いて当初のタスクを再開する。本発明の一実施例では、コントローラ92は所定の数の同時書き込みタスクをキューアップする。さらに、一実施例においては、コントローラ92は利用可能なフリースペースと、ホット/コールドブロックサイクル不均衡を求めて主フラッシュメモリアレイ96を監視し続ける。ハードウェアコントローラ92は、また、フラッシュメモリアレイ96に関する再配分動作を必要に応じて自動的に開始する。再配分動作はフラッシュメモリアレイ96のブロックの間でサイクリングを均等に配分する。再配分動作はハードウェアコントローラ92により必要に応じて完全バックグラウンドタスク方式で自動的に開始される。

【0044】ハードウェアコントローラ92は、その上に、フラッシュメモリアレイ96の中で記憶のためのスペースを解放するために、フラッシュメモリアレイ96に関するクリーンアップ動作を自動的に開始する。クリーンアップ動作はハードウェアコントローラ92により必要に応じて開始され、完全バックグラウンドタスク方式で実行される。クリーンアップ動作については以下にさらに詳細に説明する。

【0045】ファイルシステム制御のためのハードウェアコントローラ92が利用可能なフリースペースと、ホット/コールドブロックサイクルの不均衡とを求めてフラッシュメモリアレイ96を監視し続けると共に、コントローラ92がクリーンアップ動作、すなわち、再配分動作を自動的に開始するようにするためには、フラッシュ

メモリレイ96は(1)パーソナルコンピュータシステム70内部の記憶装置として存在するか、あるいは、(2)フラッシュメモリレイ96が着脱自在であるならば、パーソナルコンピュータシステム70に物理的に確実にインタロックしているべきである。フラッシュメモリレイ96が着脱自在である代替実施例の場合、パーソナルコンピュータシステム70は、フラッシュメモリレイ96に関連するバックグラウンドタスクが発生しているときに点灯するインジケータとしての発光ダイオード(「LED」)を含む。パーソナルコンピュータシステム70のユーザーは、LEDが点灯しているとき、すなわち、バックグラウンドタスクが発生しているときにはフラッシュメモリレイ96を取り除いてはならないことを知らされるのである。

【0046】一実施例では、コントローラ92は、パーソナルコンピュータシステム70への給電が停止した場合にコントローラ92にバックアップ電力を供給する小型バッテリーに結合している。これにより、コンピュータシステムへの給電が停止したとしても、バッファRAM102からフラッシュメモリレイ96へのデータ転送は確保される。ところが、本発明の別の実施例はコントローラ92のバッテリーバックアップを含んでいない。

【0047】図2〜図4に示す実施例の場合、フラッシュメモリレイ34、64及び96を固定ディスクドライブ又はフロッピーディスクドライブのいずれかの代わりに使用できることがわかるであろう。好ましい実施例においては、主フラッシュメモリレイに対するインタフェース回路は(1)入出力(I/O)マッピング形、(2)ページングメモリマッピング形、又は(3)直接メモリマッピング形のいずれであっても良い。図5は、インタフェース回路のI/Oマッピング方式を示す。

【0048】図5に示すI/Oマッピング方式は、フラッシュメモリレイ81及び局所バッファRAM83に対しマッピングする直列転送I/Oポートから構成されるI/Oブレン85を使用する。図5に示すI/Oマッピング方式は1つの主メモリブレンではなく、別個のI/Oメモリブレン85を使用するので(主メモリをシステムRAMともいう)、図5に示すI/Oマッピング方式はホストコンピュータの主メモリスペースを全く消費しない。すなわち、I/Oマッピング方式はホストコンピュータのシステムRAMを全く消費しないのである。しかしながら、直列転送I/Oポート85に結合するI/O装置(図示せず)がランダムアクセス装置ではなく、直列であるとすれば、それらのI/O装置は中央処理装置による直接実行を支援することができない。そのようなI/O装置はコードファイルを実行に備えてシステムRAMにダウンロードしなければならない。

【0049】図6は、ページングマップフラッシュメモリレイ105に対するページングメモリマッピング形

インタフェースを示す。図6に示す主メモリブレン89は主システムRAM99と、ページングウィンドウ97と、BIOS/ROM拡張ソフトウェア95と、BIOSソフトウェア93と、拡張システムRAM91とを含む。本発明の一実施例では、主メモリブレン89は1メガバイトのアドレス範囲制約101を有する。主メモリブレン89に1メガバイトのアドレス範囲制約が存在しているということは、1メガバイトのアドレス範囲制約101を超える主メモリブレン89の部分をアドレッシングするためにCPU又はマイクロプロセッサの保護モードを使用することを意味する。言い換えれば、主メモリブレン89の拡張システムRAM領域をアドレッシングするために、CPUの保護モードを使用することになる。

【0050】初期の典型的な低性能マイクロプロセッサは保護モードを含んでいなかった。また、パーソナルコンピュータシステムの初期の、性能の悪いオペレーティングシステムは、保護モードアドレッシングを処理する能力を備えていなかった。従って、そのような性能の低いマイクロプロセッサやオペレーティングシステムは、1メガバイトのアドレスを越える領域をアドレッシングすることができなかった。しかし、その後の世代の高性能のマイクロプロセッサやオペレーティングシステムの中には、1メガバイトのアドレスを越える領域をアドレッシングできるものもある。図6の場合、フラッシュメモリレイ105は小さなページ107、109及び111から構成されており、フラッシュメモリレイ105は局所バッファRAM113に結合している。本発明の別の実施例では、フラッシュメモリレイ105を任意の数のページから構成することが可能であろう。図6のページングウィンドウ97は、フラッシュメモリレイ105のページ107、109及び111へのホスト中央処理装置による直接アクセスを可能にするページングメモリマッピング形インタフェースである。各ページは、必要に応じて、フラッシュメモリレイ105の異なるセグメントを指示するように変更される。図6は、装置ドライバの拡張として使用されるページングファームウェア103をさらに示している。ページングファームウェア103はページ115、117、119及び121を含む。本発明の一実施例では、ページングウィンドウインタフェース97を使用して、ページングファームウェア103を指示する。ページングウィンドウインタフェース97によりページングファームウェア103を指示させることによって、さらに大きなファームウェア記憶スペース103への拡張が可能になる。その結果、コンピュータシステムは、1つの(通常、2キロバイト)BIOS/ROM拡張場所95を越える追加記憶スペースを得る。

【0051】図7は、主フラッシュメモリレイに対する直接メモリマッピング形インタフェースを示す。図7

の場合、主メモリブレン135は、主システムRAM137、BIOS/ROM拡張ソフトウェア139、BIOSソフトウェア141及び拡張システムRAM145に加えて、フラッシュメモリアレイマップ149と、局所バッファRAM147とを含む。図7に示す主メモリブレン135は、1メガバイトのアドレス範囲制約143を有する。フラッシュメモリアレイマップ149と、局所バッファRAMマップ147は、メモリブレン135の1メガバイトのアドレス範囲制約143を越えるところにある。すなわち、フラッシュメモリアレイマップ149及び局所バッファRAMマップ147のアドレッシングには、CPU又はマイクロプロセッサの保護モードを使用するのである。ただし、CPUは主メモリブレン135の主システムRAM137と、BIOS/ROM拡張ソフトウェア139と、BIOSソフトウェア141については、保護モードを使用せずにアドレッシングできる。フラッシュメモリアレイマップ149は、パーソナルコンピュータシステムのフラッシュメモリアレイに対して直接マッピングする。従って、フラッシュメモリアレイマップ149はパーソナルコンピュータシステムのフラッシュメモリ記憶アレイと同じ大きさである。さらに、局所バッファRAMマップ147はパーソナルコンピュータシステムの局所バッファRAMに対して直接マッピングされる。従って、局所バッファRAMマップ147はパーソナルコンピュータシステムの局所バッファRAMと同じ大きさである。

【0052】このように、直接メモリマッピング方式は大量の主メモリスベースを使用する。加えて、直接メモリマッピング方式は、1メガバイトのアドレス範囲制約143を越える主メモリブレンの部分を実行できるCPU及びオペレーティングシステムを必要とする。直接マッピング方式によれば、フラッシュメモリアレイの全体から直接にコードを実行できる。これに対し、ページングマッピング方式の場合には、フラッシュメモリアレイの一部からコードを直接実行する。直接マッピング方式は、1メガバイトの制約を越えるアドレッシングを実行可能なCPUとオペレーティングシステムを必要とする。これに対し、ページングマッピング方式は、1メガバイトの制約を越えるアドレッシングが不可能であるCPU及びオペレーティングシステムと共に働く。

【0053】図8は、フラッシュメモリアレイ401のファイル構造400を示す。このファイル構造400は、複数のセクターを含み、ディスクエミュレーションを実現するファイル構造である。ファイル構造400をセクター分割ファイル構造ともいう。ファイル構造400のようなセクター分割ファイル構造においては、コード又はデータをセクター単位で記憶する。各セクターの長さは一定であり、ファイル構造400の場合、各セクターは512個の物理バイトから構成される。セクター

は、パーソナルコンピュータシステムにおける従来のハードディスクやディスクのセクターに類似している。従来のハードディスクの場合には、トラックごとに、通常、17個のセクターがある。

【0054】MS-DOSオペレーティングシステムの場合、クラスタは論理的にアクセス可能な最小の記憶単位である。従来のハードディスクの中には、クラスタごとに4つのセクターをもつものがある。ハードディスク及び高密度の3.5インチディスクについては、1つのクラスタは1つのセクターを含む。図8のファイル構造400においては、各クラスタは割当て装置（「AU」）である。ファイル構造400では、1つのクラスタは1つのセクターから構成されている。本発明の別の実施例では、各クラスタは2つのセクターから構成されている。さらに別の実施例によれば、各クラスタは3つ以上のクラスタから構成されている。さらに別の実施例（以下に説明する）においては、クラスタではなく、論理セクターがファイル構造の割当て単位である。

【0055】図8のファイル構造400のセクター分割ファイル構造は従来のディスクファイル構造をエミュレートしたものであり、ハードディスクドライブやフロッピーディスクドライブで見られるセクター分割ファイル構造に類似している。ファイル構造400のセクター分割ファイル構造により、クリーンアップのために、消去ブロックを無作為に選択することができる。これは、このようなファイル移動が不要であるときにファイルの動きを減らす働きをする。また、過剰なサイクリングを減少させるのにも役立つ。ところが、セクター分割ファイル構造はコードファイルの代わりに完全な連続する実行を支援することはできない。これに対し、連続セクターページングメモリファイル方式は、その代わりに、大きな連続する部分又はより小さなコードファイルの実行を可能にする。セクター分割ファイル構造の場合、クリーンアップ中にホット/コールド消去ブロック再配分の活動管理を支援するために、個々のブロックのサイクリングを追加しなければならない。このような個々のブロックのサイクリングは、本質的には、以下に論じるクリーンアップ/再割当て規則により管理される。

【0056】フラッシュメモリアレイ401は、コード又はデータを記憶する多数のデータブロックを含む。ここで使用する「データ」という用語は、実行可能コード又は非実行コード、もしくはその双方を含むものとする。図8は、クリーンアップ動作直後の状態にあるフラッシュメモリアレイ401のファイル構造の1例を示す。フラッシュメモリアレイ401のブロック416、424及び428は、予約ブロック又は予約済ブロックとも呼ばれる「予約すべき」ブロックである。予約ブロックは、クリーンアップ動作中に一時ファイルバックアップを行うブロックである。

【0057】本発明の一実施例では、(1)利用可能な

フラッシュ記憶容量がある（構成可能）限界が低下した場合、（２）パーソナルコンピュータシステムのRAMの中止・猶予常駐プログラムなどの特殊クリーンナップユーティリティプログラムによりクリーンナップを実行するときに、ユーザーが明示してクリーンナップを要求した場合、又は（３）クリーンナップをトリガする（構成可能）時間間隔が経過した場合に、クリーンナップ動作をトリガすることができる。クリーンナップ動作を再利用サイクルともいう。クリーンナップ動作は、第１に、再割当て動作を含む。再割当て動作中、現在活動中の全てのファイルを、最終的に消去されるブロックから取り除く。それらの現在活動ファイルを予約ブロックへ移動する。消去すべき各フラッシュメモリブロックを全て一度に消去する。ファイル構造４００の場合、クリーンナップ動作中、再割当てのために予約ブロック４１６、４２４及び４２８を利用できる。従って、予約ブロック４１６、４２４及び４２８はファイルの一時バックアップを行う。

【００５８】ブロック４０８は、次のクリーンナップ動作の後に空きブロックになる空き予定ブロックである。ブロック４０４、４１４及び４２０は、コード及びデータを記憶する使用可能ブロックである空きブロックである。図８に示す時点で、空きブロック４０４、４１４及び４２０は活動ブロックではない。その代わりに、ブロック４０４、４１４及び４２０は将来コード又はデータを記憶するために使用すべきブロックである。図８に示す時点では、活動ファイルに関わるデータとコードはブロック４０２、４０６、４１２、４１８、４２６及び４３０の活動セクターに記憶される。パーソナルコンピュータシステムがさらに動作を続ける間に、データ又はコードは空きブロック４０４、４１４及び４２０のセクターに導入される。

【００５９】データブロック４３０は一部空きブロックである。ブロック４３０の一部分４６４は、活動セクターに記憶されるコード又はデータを含んでいる。ブロック４３０の部分４６２はコード又はデータを記憶するために利用できる空きスペースである。パーソナルコンピュータシステムがさらに動作を続ける間に、コードとデータはブロック４３０の部分４６２にも導入される。空きブロックや、部分空きブロックの中のセクターがコード及びデータで充填されるにつれて、データ又はコードを記憶しないままであるブロックの数は減少する。使用可能な空きブロックの数が予約ブロックの許容数に達すると、クリーンナップ動作は開始する。

【００６０】ブロック４０２、４０６、４１２、４１８及び４２６の中のセクターは、ブロック４３０の部分４６４の中のセクターと共に、活動ファイル及び削除ファイルと、サブディレクトリとを構成する。ブロック４０２の中のセクター４０３、４０５、４０９、４１１、４１３、４１５、４１７、４２３、４２５及び４２７と、

ブロック４０６の中のセクター４３１、４３３、４３９、４４３、４４５及び４５３とは、活動ファイル及びサブディレクトリセクターの一例である。ブロック４１２、４１８、４２６及び４３０も活動ファイルと、サブディレクトリセクターとを含む。セクター４０７、４１９、４２１、４２９、４３５、４３７、４４１、４４７、４４９、４５１、４５７、４５９、４６１、４６３、４６５、４６７、４６９、４７１、４７３、４７５、４７７、４７９、４８１、４８３、４８５、４８７、４８９及び４９１は、削除ファイルと、サブディレクトリとを含むファイルセクターである。

【００６１】クリーンナップが要求されると、パーソナルコンピュータシステムは、後続する再書込みに備えてどのブロックをクリアすべきかを識別する。言いかえると、クリーンナップ時には、パーソナルコンピュータシステムは、クリーンナップ時に再割当てし、移動すべきファイルを有するブロックを識別する。クリーンナップ時にクリアすべきブロックの識別を管理する規則は２つある。第１の規則は、サイクリングが最も少ないブロックを選択することである。これにより、ホットブロックとコールドブロックとのサイクリングの不均衡は減少する。ブロックのサイクルカウントに大きな不均衡が存在しないならば、第２の規則が主となる。言い換えれば、サイクリングの配分がパーソナルコンピュータシステムにより問題としてフラグ付けされない場合には、第２の規則が主となることになる。第２の規則は、削除セクターの数が最も多いブロックがクリアすべきブロックとして選択されるブロックであることを定める。

【００６２】この第２の規則を採用するのは、削除ファイル又は削除サブディレクトリを含むセクターを消去に先立ってブロックから除去する必要があるからである。削除ファイルと削除サブディレクトリはセーブしなければならない情報を含んでいないので、削除ファイルと削除サブディレクトリをクリーンナップ動作の一部として消去することができる。すなわち、削除ファイル又は削除サブディレクトリを再割当てする必要はないのである。従って、第２の規則を適用した場合、クリーンナップ動作の一部である消去の前に実行される活動セクターの除去の量は最小限に抑えられる。

【００６３】どのブロックが最も多い削除ファイルセクターを含むかを管理するのは本質的に無作為のプロセスである。従って、第２の規則は、クリーンナップ方式によって、クリアすべきブロックに関して主として無作為であるブロック選択が行われることを意味している。この本質的に無作為のブロック選択はクリーンナップ効率を向上させるのを助けると同時に、全体的なサイクリング不均衡をできる限り少なくする。上記の第１の規則と、第２の規則は、共に、フラッシュメモリアレイ４０１の内部におけるサイクリングの配分を管理する。第１の規則と第２の規則は共にサイクリングの差異を最小に

するのに役立つ。そのため、ファイル構造400のクリーンナップは本質的に無作為になるのである。

【0064】3つのブロック406、418及び426を予約ブロック416及び424にはめ込むことができるように、クリーンナップ動作中に圧縮動作が起こる。圧縮動作中、先にユーザーが削除したファイルはブロック416及び424に再書き込みされない。圧縮動作を短縮ともいう。図8のブロック406、418及び426の中にあるセクター429、435、437、441、447、449、451、463、465、467、469、471、473、475、477、479、481、483、485、487、489及び491は削除ファイルと、削除サブディレクトリを含む。それらの削除セクターはブロック416及び424に再書き込みされない。ブロック406、418及び426は圧縮すべき次の領域に存在するという。クリーンナップ後、ブロック416及び424は圧縮スペースの一部を形成することになる。ブロック406、418及び426に関する圧縮自在スペースの量は、削除セクター429、435、437、441、449、451、463、465、467、469、471、473、475、477、479、481、483、485、487、489及び491が占めているスペースの量である。ブロック406、418及び426をブロック416及び424に圧縮した後、ブロック406、418及び426を消去する。ブロック406、418及び426は、消去後、予約ブロックとなる。ブロック428は予約ブロックのままである。ブロック408は空きブロックのままである。

【0065】ファイル構造400は、論理クラスタをファイル構造400の物理セクターにマッピングするクラスタマッピングテーブル446を使用する。クラスタマッピングテーブル446はファイル構造400の物理セクターと、外界とを連係させる。クラスタマッピングテーブル446は、ファイル構造400を、フラッシュメモリアレイ401が隣接セクターを伴う従来の固定ディスク又はフロッピーディスクであるかのように外界に対して見せる。当然のことながら、現実には、フラッシュメモリアレイ401がブロック単位でのみ消去できると共に、ファイル構造400がクリーンナップ動作を含むものとすれば、フラッシュメモリアレイ401は従来の固定ディスク又はフロッピーディスクとは異なる。クラスタマッピングテーブル446は、パーソナルコンピュータシステムにより要求される論理セクターをフラッシュメモリアレイ401の物理セクターにマッピングすることにより、それらの差を克服する。クラスタマッピングテーブル446はフラッシュメモリアレイ401のブロック422の中にある。このブロック422をクラスタマッピングブロック又はフラッシュディスクエミュレーションブロックという。ブロック422はクラスタマ

ッピングテーブルフラグ440と、フラッシュディスクエミュレーション(「FDE」)ブート記録442と、フラッシュディスクエミュレーション構成ブロック状態テーブル444と、空きスペース448とをさらに含む。空きスペース448は、クラスタマッピングテーブル446の拡張のための余地を形成する。

【0066】クラスタマッピングテーブルフラグ440は、どのブロックがファイル構造400に関わるクラスタマッピングテーブルを記憶しているかを指示するデータのパターンを記憶する。クラスタマッピングテーブル(「CMT」)フラグ440は、CMTフラグ440を含むブロックがクラスタマッピングテーブルを含むブロックでもあることを示す一意の非無作為データパターンを記憶する。たとえば、一実施例では、CMTフラグ440は一連の反復AAh/55hデータを記憶している。別の実施例においては、CMTフラグ440は他の何らかの非無作為データシーケンスを記憶している。本発明の一実施例においては、実際に特定の1つのブロックがクラスタマッピングブロックであるか否かを判定するために、補足検査を実行する。この検査は、クラスタマッピングブロック以外のブロックが、CMTフラグ440のデータのパターンと一致するデータの開始パターンを有する場合に起こりうるクラスタマッピングブロックの位置決め誤りを回避するのを助けるために実行される。補足検査は、クラスタマッピングブロック422のフラッシュディスクエミュレーションブート記録442に存在している検査合計データフィールドを検査することにより実行される。一実施例では、ブート記録442の検査合計データフィールドは、そのFDEブート記録442が有効であり且つ現在のものであれば変化しないデータのパターンであるので、選択された特定のパターンである。別の実施例においては、FDEブート記録442の検査合計データフィールドは、専用サイクル冗長検査(「CRC」)又は誤り修正コード(「ECC」)データフィールドである。まず、コンピュータシステムはCMTフラグを探索する。第2に、コンピュータシステムは検査合計データフィールドを検査する。FDEブート記録442の検査合計データフィールドの検査から得られる正しい結果は、コンピュータシステムが実際にフラッシュメモリアレイ401について現在有効クラスタマッピングブロックを発見したことをコンピュータシステムに知らせる。

【0067】図8に示すように、フラッシュディスクエミュレーションブート記録442はブロック422の中の、CMTフラグ440の次に位置している。FDEブート記録442はファイル構造400の特定のファイルシステム型と、特定のファイルシステム改訂とに関する情報を含む。FDEブート記録442はいくつかのファイルシステムパラメータをさらに含む。それらのパラメータはフォーマット作成可能容量情報と、予約ブロック

情報と、付属クラスタマッピングテーブル情報と、旧クラスタマッピングテーブルに関連する情報と、現在クラスタマッピングテーブルに関連する情報とを含む。フォーマット作成可能容量情報はフラッシュメモリアレイ401のうち、コード及びデータを記憶するためにフォーマット作成することができる領域の大きさである。予約ブロック情報は、どのブロックが予約ブロックであるかを識別する。前述の通り、予約ブロックはクリーンアップ動作及び再割当て動作の間に使用されるブロックである。付属クラスタマッピングテーブル情報は、付属クラスタマッピングテーブル情報を記憶するために使用されている追加ブロックの場所である。一実施例では、クラスタマッピングテーブル記述項の数がある量を越えた場合に、1つ又は2つ以上の追加ブロックに記憶されている付属クラスタマッピングテーブルを使用する。旧CMTは現在クラスタマッピングテーブルのバックアップコピーである。旧クラスタマッピングテーブルに関連するパラメータ情報は、フラッシュメモリアレイ401の中の、その旧CMTが記憶されている場所である。たとえば、図8の場合、旧CMT445を記憶しているのはブロック410である。クラスタマッピングテーブルに関連するパラメータ情報は、フラッシュメモリアレイ401の中のクラスタマッピングテーブル446の場所である。

【0068】FDEブロック状態テーブル444は、どのブロックが活動ブロックであるか、どのブロックが予約ブロックであるか、及びどのブロックが欠陥ブロック又は故障ブロックであるかというに関する情報を含む。さらに、FDEブロック状態テーブル444はファイル構造400の各ブロックについてのサイクルカウント情報を含む。ブロック状態テーブル444は、後続のクリーンアップを可能にするために、クラスタマッピングブロック422及び予約ブロックをアクセス不可能として留保しておく。これにより、コンピュータシステムがそれらのブロックに重ね書きしようとする試みは阻止される。故障ブロックも回避され、ブロック状態テーブル444の中に指示される。FDEブロック状態テーブル444はブロック422の中の、FDEブート記録442の次に位置している。

【0069】クラスタマッピングテーブル446はブロック422の中の、FDEブロック状態テーブル444の次に位置している。クラスタマッピングテーブル446は、パーソナルコンピュータシステムのオペレーティングシステムが送信する論理クラスタアドレスをフラッシュメモリアレイ401の中の物理アドレスにマッピングする。クラスタマッピングテーブル446は、フラッシュメモリアレイ401及びその関連ファイル構造400を固定ディスクドライブ又はフロッピーディスクドライブであるかのようにパーソナルコンピュータシステムに見せることができる。

【0070】図9、図10及び図11は、クラスタマッピングテーブル446の編成と動作を示す。クラスタマッピングテーブル446は、図9に示す論理クラスタアドレスを欄521に列挙した物理アドレスと関連させるテーブルである。この相関を連関マッピングともいう。リスト520に見られる論理クラスタアドレスは、パーソナルコンピュータシステムのディスクオペレーティングシステムがそのパーソナルコンピュータシステムに関わるコード又はデータを記憶する固定ディスクドライブ又はフロッピーディスクドライブに送信するであろうと考えられる論理アドレスである。従来のコンピュータシステムの場合、論理クラスタアドレスは固定ディスクドライブ又はフロッピーディスクドライブのクラスタのアドレスである。各クラスタは、従来の固定ディスク又はフロッピーディスクのトラックを構成している1つ又は複数のセクターから形成される。クラスタごとのセクターの数は、従来の固定ディスクドライブ又はフロッピーディスクドライブの大きさ又は構成によって決まる。

【0071】言うまでもなく、フラッシュメモリアレイ401は従来の固定ディスク又は従来のフロッピーディスクではない。フラッシュメモリアレイ401は、相対的に大きなブロック単位でのみ消去可能なフラッシュメモリから構成されている。さらに、フラッシュメモリアレイ401の個々のビットセルをあらかじめ消去せずに論理値0から論理値1に重ね書きすることは不可能である。図8に示す実施例の場合、ファイル構造400はクラスタごとに1つのセクターを有する。ファイル構造400においては、順次データを含むセクターを異なるブロックの間に分散させても良い。たとえば、コードの最初の行が、図8のフラッシュメモリアレイ401のブロック402のセクター409にあり、同じコードの終わりの行はブロック406のセクター445にあってても良い。従来の固定ディスクドライブ又はフロッピーディスクドライブでは、先に消去サイクルを経なくとも、データのビットを重ね書きすることができる。これに対し、図8のフラッシュメモリアレイ401の場合には、あらかじめ消去せずに個々のビットセルを論理値0から論理値1に重ね書きすることは不可能である。従って、フラッシュメモリアレイ401については、(1つのビットを論理値1から論理値0に書込む場合を除いて)あらかじめ消去を実行することなく新たなデータは旧データに重ね書きされない。その代わりに、旧セクターを汚れたセクターとして識別する。これは、そのセクターがそれ以上使用すべきではないデータを含むことを表す。そこで、汚れたセクターをクリーンアップ動作中に消去する。すなわち、汚れたセクターは次のクリーンアップ動作のときに消去すべきデータ又はコードを含んでいるのである。たとえば、図8に示すファイル構造400のセクター407は、望ましくない旧コード又は旧データを含む汚れたセクターである。

【0072】新たなデータ又はコードを汚れたセクター407に重ね書きする代わりに、単に、フラッシュメモリアレイ407の中の空きブロック又は予約ブロックにある利用可能スペースに新たなコード又はデータを書込む。たとえば、予約ブロック416の利用可能セクターの中の1つに新たなデータ又はコードを書込むのである。

【0073】フラッシュメモリアレイ401の特性と、従来の固定ディスクドライブ又はフロッピーディスクドライブの特性には差があるにもかかわらず、クラスタマッピングテーブル446は図8のフラッシュメモリアレイ401を、それが固定ディスクドライブ又はフロッピーディスクドライブであるかのようにパーソナルコンピュータシステムのオペレーティングシステムに見せることができる。クラスタマッピングテーブル446は、クラスタ論理アドレスをフラッシュメモリアレイ401の中の物理セクターにマッピングすることにより、これを実行する。クラスタマッピングテーブル446は、複数の係リスト記述項から構成されるテーブルである。クラスタマッピングテーブル446が係リストを含んでいるため、クラスタマッピングテーブルを時間の経過につれて追加してゆくことができる。

【0074】図9に示すクラスタマッピングテーブル446はリスト520中に見られる論理アドレスを欄521の物理アドレスにマッピングしている。欄522は、テーブル記述項ごとのポインタを示す。欄522の特定の1つの記述項に記憶される全てが論理値「1」の空きパターンは、その特定の記述項が係リストの終わりであることを指示する。全て論理値1の空きパターンをF空(16進数)データパターンという。図9に示すクラスタマッピングテーブル446の実施例の場合、クラスタごとのセクターは1つだけである。CMT446は次のように動作する。クラスタマッピングテーブル446の記述項551はクラスタ論理アドレス0をフラッシュメモリアレイ401の物理セクターアドレスにマッピングする。説明をわかりやすくするため、図9の欄522に示した物理アドレスは図8のファイル構造400のセクターに対する図中符号と同じにしてある。従って、記述項551の物理アドレス403は図8のファイル構造400のセクター403に対応する。図9の記述項551の場合、欄522のポインタはF空データパターンを含んでいる。これは、論理アドレス0については他に係リスト記述項が存在しないことを表わす。すなわち、ファイル構造400のセクター403は汚れたセクターではないのである。これは、物理アドレス403に見られるセクターは実際には論理クラスタアドレス0に対応することを意味している。

【0075】同様に、クラスタマッピングテーブル446の記述項553は、クラスタ論理アドレス1をファイル構造の物理アドレス405にマッピングしている。物

理アドレス405は図8に示すセクター405に対応する。記述項553は次の論理アドレスにF空データパターン、すなわち空の欄522を含んでいる。これは他に係リスト記述項が存在しないことを示しているので、物理アドレス405はクラスタ論理アドレス1に対応する。クラスタマッピングテーブル446の記述項555は係リストの最初の記述項である。記述項555はセクター論理アドレス2を物理アドレス407にマッピングしている。物理アドレス407にあるのはフラッシュメモリアレイ401のセクター407である。フラッシュメモリアレイ401のセクター407は汚れたセクターである。すなわち、セクター407は(1)ブロック402に関わる次のクリーンアップの時に削除すべき旧情報を含むか、又は(2)物理的な問題を有し、データ記憶のためにはそれ以降使用されない不良セクターであるかのいずれかである。不良セクターの1例としては、欠陥のあるフローティングゲートセルを1つ又は2つ以上含み、そのために、誤りなくデータを記憶することが不可能であるセクターが考えられるであろう。

【0076】記述項555に関する欄522の次の論理アドレス又は空記述項はアドレスSを指示している。アドレスSは、記述項555についてポインタにより指示される再マッピング(すなわち、付属)論理アドレスである。図9では、文字Sはフラッシュメモリアレイ401の区分ごとの論理セクターの数も表わしている。さらに、Sはファイル構造400がエミュレートしている仮定上の固定ディスク又はフロッピーディスクの区分ごとの論理セクターの数に等しい。論理アドレスSはクラスタマッピングテーブル446の記述項563に対応する。記述項563の場合、欄521に物理アドレス421が記述されている。従って、フラッシュメモリアレイ401の物理アドレス421は再マッピング論理アドレスSに対応する。物理アドレス421にはフラッシュメモリアレイ401のセクター421が見られる。セクター421は汚れたセクターである。クラスタマッピングテーブル446の記述項563に関わる次の論理アドレス記述項はアドレスS+1を含む。クラスタマッピングテーブル446の記述項565は再マッピング論理アドレスS+1を物理アドレス449にマッピングする。物理アドレス449はフラッシュメモリアレイ401のセクター449に対応する。図8のセクター449は汚れたセクターである。記述項565に関わる次の論理アドレス記述項は論理クラスタアドレスS+2である。クラスタマッピングテーブル446の記述項567は論理付属クラスタアドレスS+2を物理アドレス443にマッピングする。クラスタマッピングテーブル446の欄521の物理アドレス443は、図8に示すセクター443に対応する。セクター443はファイル構造400のブロック406に見られる。セクター443は汚れたセクターではなく、問題のない活動セクターである。図9

の記述項567に関わる次の論理アドレス又は空記述項はF空(16進数)記述項である。これは、論理クラスタアドレス2と共に始まる連係リストの終わりを指示する。

【0077】図10は、クラスタマッピングテーブル446の記述項555、563、565及び567により形成される連係リストを示す。記述項555は論理クラスタアドレス2を物理セクタアドレス407にマッピングしている。555の「次の」ポインタ記述項は記述項563を指示している。記述項563の「次」ポインタは記述項565を指示している。記述項565の「次」ポインタは記述項567を指示している。記述項567のポインタはF空(16進数)データパターンを含む。この空データパターンは、記述項567が連係リストチェーンの最終記述項であることを示す。図9に関して述べた通り、セクタ407、421及び449は全て汚れたセクタである。セクタ443は「きれいな」、すなわち、優良な活動セクタである。それは、連係リストチェーンがセクタ443と共に終わるからである。言い換えれば、連係リストチェーンはその連係リストチェーン内部の最初の活動優良セクタと共に終わる。図10に示す連係リストの最終結果は、クラスタマッピングテーブル446が論理クラスタアドレス2を物理セクタアドレス443にマッピングすることになり、物理セクタ443は「きれいな」活動セクタである。

【0078】汚れたセクタが作成又は除去されるたびに、クラスタマッピングテーブル446中の連係リストチェーンに1つのリンクが追加される。すなわち、クラスタマッピングテーブル446の中の各連係リストチェーンは汚れたセクタの作成と除去の履歴記録を表わすのである。たとえば、記述項555、563、565及び567から形成される連係リストチェーンの場合、セクタ407は一時は有効データ又は有効コードを含むきれいな活動セクタであった。その以前の時点では、CMT446は論理アドレス2を物理アドレス407にマッピングしており、記述項555に関わるポインタはF空データパターンを含んでいた。ところが、パーソナルコンピュータの動作中のいずれかの時点で、セクタ407は汚れたセクタとなった。セクタ407が汚れたセクタになってしまうと、記述項555に関わるポインタは全て1のF空パターンから、付属論理アドレスSを2進数で表わす新たなパターンに変わる。このF空パターンからアドレスSへの変更は、フラッシュメモリアレイ401の場合には個々のビットセルを論理値1から論理値0に重ね書きできるために、重ね書きすることにより実行される(あらかじめ消去せずに)。F空パターン(全て論理値1から成る)を、アドレスSを作成するために1つ又は2つ以上のビットを論理値1から論理値0に重ね書きすることにより、アドレスSに変更し

た。

【0079】次に、記述項563を再マッピング(すなわち、付属)記述項としてクラスタマッピングテーブル446に記憶した。再マッピングアドレスSをクラスタマッピングテーブル446において物理アドレス421にマッピングした。物理アドレス421はセクタ421に対応する。先の時点では、セクタ421は汚れたセクタではなく、優良なセクタであった。その時点では、記述項563に関わるポインタはF空データパターンを含んでいた。従って、その時点で、クラスタマッピングテーブル446は論理アドレス2をセクタ421にマッピングした。時間の経過につれて、セクタ421は汚れたセクタとなった。そこで、論理クラスタアドレス2について連係リストに別のリンクを追加する必要がある。従って、記述項563に関わるポインタを空データパターンから再マッピングアドレスS+1に変更した。次に、付属アドレスS+1を物理アドレス449にマッピングするために、記述項565をクラスタマッピングテーブル446に記憶した。物理アドレス449はセクタ449に対応する。その時点における記述項565に関わるポインタはF空データパターンを含んでいた。その時点では、セクタ449は優良セクタであり、汚れたセクタではなかった。従って、その時点で、クラスタマッピングテーブル446は論理アドレス2を物理アドレス449にマッピングした。それにより後の時点で、セクタ449は汚れたセクタになった。この事態が起こると、記述項565に関わるポインタはF空パターンからアドレスS+2に重ね書きされる。次に、記述項567をクラスタマッピングテーブル446に記憶した。記述項567は「次の論理アドレス又は空」の欄522にF空データパターンを含んでいた。そこで、クラスタマッピングテーブル446は図9に示す通りの形となる。

【0080】先に述べたように、論理アドレスS-1は最終論理クラスタアドレスである。図9に示すSからXまでの再マッピング(すなわち、付属)アドレスは、全て、連係リストを形成するために使用されるアドレスである。記述項571は最終論理アドレスXを物理アドレス443にマッピングし、そのポインタとしてF空データパターンを含んでいる。図9を参照すると、クラスタマッピングテーブル446はその他にも記述項を含む。クラスタ記述項561は論理クラスタアドレスS-1を物理アドレス453にマッピングする。記述項561に関わるポインタはF空データパターンである。記述項571の場合、付属アドレスXはF空データパターンを伴うポインタを有する。

【0081】図11は、クラスタマッピングテーブル446がどのように動作するかを示す。ディスクオペレーティングシステムはクラスタマッピングテーブル446の中に記憶されている論理クラスタアドレス20を見



る。クラスタマッピングテーブル446はそれらの論理アドレス520をフラッシュメモリアレイ401内部のフラッシュセクターにマッピングする。前述のように、ファイル構造400においては、クラスタごとに1つのセクターがある。図11に示すように、クラスタマッピングテーブル446はクラスタ0からS-1をマッピングしている。クラスタS-1が最終クラスタとなるのは、クラスタが論理アドレス0から始まり、区分ごとにS個のクラスタが存在しているためである。論理クラスタ0はフラッシュメモリアレイのセクター403にマッピングされ、論理クラスタ1はフラッシュメモリアレイ401のセクター405にマッピングされている。フラッシュセクター403及び405はフラッシュメモリアレイ401のフラッシュブロック402の中にある。フラッシュセクター403及び405は共に活動優良セクターである。図11に示す通り、当初、論理クラスタアドレス2はクラスタマッピングテーブル446によってフラッシュメモリアレイ401のフラッシュセクター407にマッピングされていた。その後、フラッシュセクター407は汚れたセクターとなった。汚れたフラッシュセクター407はフラッシュブロック402にある。その後、クラスタアドレス2について、クラスタマッピングテーブル446の中に連係リストを作成するためにリンクを追加した。そのような連係リストを作成した結果、論理クラスタ2はフラッシュブロック446の中のフラッシュセクター443にマッピングされることになる。

【0082】連係リストの中にはいくつかのリンクがあっても良いが、最終結果は、当初、クラスタ2がフラッシュセクター407にマッピングされていたとしても、新たにフラッシュセクター443にマッピングされるということである。一実施例では、2ギガバイトまでのフラッシュメモリアレイ401をアドレッシングすることができる。別の実施例によれば、アドレッシング可能な最大スペースは2ギガバイトより大きくても、小さくても良い。

【0083】図9は、クラスタマッピングテーブル446の1例を示そうとするものであり、他の変形も可能である。さらに、クラスタマッピングテーブル446は論理クラスタアドレス、付属アドレス及び記述項の一部しか示していない。アドレス2からアドレスS-1までの間並びに付属アドレスS+2から付属アドレスXまでの間には、数多くの記述項や論理アドレスが見られる。本発明の別の実施例においては、ファイル構造400はクラスタごとに2つ（又は3つ以上）のセクターを有する。クラスタはこの場合にも割当て単位である。その実施例の変形によれば、どの特定の論理クラスタについても、その物理セクターがフラッシュメモリアレイ401の中で互いに物理的に順次並ぶように、物理セクターをフラッシュメモリアレイに書込む。言い換えれば、クラ

スタごとの物理セクターは物理的に1つのグループにまとめられ且つ隣接している。この実施例の場合のクラスタマッピングテーブルは論理クラスタアドレスごとに2つ（又は3つ以上）の隣接する物理セクターをマッピングする。別の実施例では、クラスタごとに2つ（又は3つ以上）のセクターが存在しているが、どの特定のクラスタについても、その物理セクターをフラッシュメモリアレイ401のデータ領域又はコード領域全体に分散させることができる。すなわち、その実施例の場合には、クラスタごとの物理セクターを1つのグループにまとめる必要はなく、従って、物理セクターは隣接していなくとも良いのである。この実施例のクラスタマッピングテーブルは論理クラスタアドレスごとに2つ（又は3つ以上）の物理セクターをマッピングする。本発明のさらに別の実施例によれば、ファイル構造400はクラスタマッピングテーブル446に代わるセクターマッピングテーブルを含む。このセクターマッピングテーブルはセクターの論理アドレスをセクターの物理アドレスにマッピングする。この実施例の場合、論理セクターが割当て単位である。その他の点については、セクターマッピングテーブルはCMT446と同じように動作する。その場合、セクターマッピングテーブルは、たとえば、ブロック422のようなセクターマッピングブロックの中にある。この実施例では、CMTフラグ440及びCMTスペース448の代わりにセクターマッピングテーブルスペースという用語を用いる。本発明の一実施例では、クラスタマッピングテーブル446のコピーをバックアップコピーとして図8のファイル構造400の中の空きセクター又は予約セクターに記憶しておく。本発明の一実施例においては、クラスタマッピングテーブル446のバックアップコピーを周期的に作成する。本発明の一実施例では、クラスタマッピングテーブル446のバックアップコピーをクリーンアップ動作が終了するたびに作成する。

【0084】一実施例においては、クラスタマッピングテーブル446のバックアップコピーはクラスタマッピングテーブル446の圧縮バージョンである。図12のテーブル650は、クラスタマッピングテーブル446の圧縮バージョンを表わす。テーブル650は欄620のクラスタ論理アドレスと、欄621の物理アドレスのみを表わす。記述項ごとのポインタと付属（すなわち、再マッピング）アドレスは含まれていない。テーブル650は論理アドレス0～S-1（Sは区分ごとのセクターの数を表わす）を含み、付属アドレスを含まない。欄621に記憶されている各アドレスは、対応するクラスタ論理アドレスと関連する活動優良セクターの物理アドレスである。テーブル650の場合、連係リストチェーンは記憶されていない。記憶されているのは連係リストチェーンのクラスタ論理アドレスの始まりと、物理アドレスの終わりと、連係リストを伴わない記述項のみであ

る。記述項651は論理クラスタアドレス0を物理アドレス403にマッピングしている。記述項653はクラスタ論理アドレス1を物理アドレス405にマッピングしている。記述項655は論理アドレス2を物理アドレス443にマッピングしている。図9と図12を比較すると、記述項555、563、565及び567の組み合わせは最終的にはクラスタ論理アドレス2を物理アドレス443にマッピングすることがわかる。それら4つの記述項555、563、565及び567は連係リストチェーンを構成している。圧縮バージョンのテーブル650はその連係リストチェーンの始まりと終わりのみを記憶しているのである。言い換えれば、記述項655はクラスタ論理アドレス2と物理アドレス443との関係を記憶しているにすぎない。最後に、テーブル650の記述項661は論理アドレスS-1を物理アドレス453にマッピングしている。

【0085】一実施例では、クラスタマッピングテーブル446又はクラスタマッピングテーブル446の圧縮バージョンをパーソナルコンピュータシステムのRAMに記憶する。コンピュータシステムには、図8のファイル構造400は、それがハードディスクドライブ又はフロッピーディスクドライブであるかのように見える。ファイル構造400には、ブート記録、ファイル割当てテーブル、ルートディレクトリ記述項、並びにディスクドライブに記憶されるとすれば有するであろう論理アドレスと同じ論理アドレスをもつデータ領域をロードすることができる。すなわち、図8に示すフラッシュメモリアレイ401は図1に関して論じたような従来のMS-DOSオペレーティングシステムなどと共に使用可能なのである。一実施例では、フラッシュディスクをエミュレートするファイル構造400は(1)DOS、(2)OS/2(すなわち、ニューヨーク州アーモックのInternational Business Machinesのオペレーティングシステム/2)及び(3)UNIXオペレーティングシステムの既存のあらゆるバージョンを支援する。フラッシュディスクエミュレーションファイル構造400はDOS、OS/2、UNIX、BIOS又は他のディスク常駐データ構造の特定のバージョンに依存しない。ファイル構造400はDOS及びOS/2の低レベルディスクユーティリティをも支援する。一実施例においては、ファイル構造400は、割込み23H及び26H並びに標準DOS装置ドライバインタフェースにより規定される読取り動作及び書込み操作のための直接ディスクアクセスを支援する。一実施例では、フラッシュディスクエミュレーションドライバ構造400は、BIOS割込み13Hにより規定される全ての機能を支援する。

【0086】一実施例では、フラッシュメモリアレイ401をMS-DOSオペレーティングシステムと関連させて使用する。図1と図8の双方を参照して説明する

と、ファイル構造400はシステム領域4と、データ領域9の2つの論理領域に分割されている。システム領域はブート記録3と、ファイル割当てテーブル5と、ルートディレクトリ記述項を記憶するルートディレクトリ領域7とを含む。データ領域9は、アプリケーションプログラムと、データと、サブディレクトリ情報とを記憶するために使用されるファイルを含む。ブート記録3、ファイル割当てテーブル5及びサブディレクトリ領域7は、論理クラスタ0から始まるファイル構造400の始まり論理クラスタを占める。この場合にも、図1に関して示す構成は論理アドレス構成である。ブート記録3と、ファイル割当てテーブル5と、ルートディレクトリ領域7と、データ領域9とを記憶するフラッシュメモリアレイ401の実際の物理セクターはクラスタマッピングテーブル446によって決定される。ブート記録3は、オペレーティングシステムをロードするためのブートストラップロードプログラムを含む。ブート記録3はフォーマット作成DOSのASCII名、セクターごとのバイト数、クラスタごとのセクター数、ブート記録に含まれるセクターの数、ファイル割当てテーブルのコピーの数、ルートディレクトリ記述項の数、区分ごとのセクター数、メモリ(すなわち、エミュレートディスク)の型番号、ファイル割当てテーブルごとのセクター数、予約セクター又はかげのセクターの数、フラッシュメモリアレイ401に関わる識別番号、拡張ブートセクターシグナチャ、ボリュームID及びボリュームテーブルに関する情報をさらに含む。ルートディレクトリ7は、それぞれがファイルのいくつかの属性を記述している複数の32バイト記述項から成るテーブルである。ルートディレクトリ7を構成する各ディレクトリ記述項はファイル名と、ファイル拡張と、属性フラグと、ファイルの日時スタンプと、ファイルを構成しているクラスタの開始クラスタ番号と、ファイルサイズとを含む。ディスクの各ファイルは1つ又は2つ以上のクラスタから構成されている。ファイル割当てテーブル5は、ファイルを構成するクラスタが互いにどのように連係しているかという連鎖の形態の記録を含む。一実施例では、FAT5はクラスタごとに1つずつある2バイト記述項のリストを含む。別の実施例によれば、FATの記述項は2バイトより長い又は短い。あるファイルのディレクトリ記述項は、そのファイルの開始クラスタ番号と、ファイル割当てテーブルをアクセスするためにその開始クラスタ番号を使用するオペレーティングシステムとを含む。FATの各記述項はファイルの次のクラスタを指示するポイントである。従って、その第1回のアクセスによって検索されるFAT記述項は、ファイルを構成している次のクラスタのクラスタ番号を含む。オペレーティングシステムは次のクラスタ番号を使用してFATをアクセスし、さらに次のクラスタ番号を検索し、FAT5の中の特別のマーカに達するまでこのプロセスを続ける。

【0087】MS-DOSの場合、ファイル構造400のファイルの論理構造は樹木形である。ルートディレクトリの記述項はサブディレクトリに対するポインタであると考えることができる。サブディレクトリは入れ子形にすることができる。コードとデータを記憶するファイルは、図8のフラッシュメモリアレイ401の中にある物理セクターの至るところに分散している。この場合にも、それらのファイルの論理アドレスを、それらのファイルを記憶しているセクターの物理アドレスにマッピングするために、クラスタマッピングテーブル446を使用する。本発明の別の実施例では、図8に示すファイル構造400はさらにヘッダを含む。たとえば、ヘッダはブロック414にあり、このブロックはその場合には空きブロックではなくなる。

【0088】ヘッダは、フラッシュメモリアレイ401に関する情報を含むファイルである。一実施例では、ヘッダはフラッシュメモリアレイ401全体のフォーマット作成していないときのサイズと、フラッシュメモリアレイ401のフォーマット作成時のサイズと、フラッシュメモリアレイ401内部のブロックの総数と、フラッシュメモリアレイ401に関連する詳細な装置情報とに関する情報を含む。本発明の一実施例においては、ヘッダに記憶される詳細な装置情報はフラッシュメモリアレイ401の消去電圧及び書き込み電圧と；消去指令及び書き込み指令と；消去アルゴリズム及び書き込みアルゴリズムと；個別チップ最大サイクリング仕様及び全チップ最大サイクリング仕様と；読取り、書き込み及び消去の性能特性を含んでいる。別の実施例では、ヘッダは1つ又は複数の代替ヘッダの場所に関する情報をさらに含む。ヘッダをフラッシュメモリアレイ401のブロックの中の1つ、たとえば、ブロック414に記憶するこの別の実施例においては、ヘッダを周期的に再生しなければならない。ヘッダの再生はヘッダを完全に消去し、再書き込みすることを周期的に実行することを伴う。通常、ヘッダはフラッシュメモリアレイ401のその他のブロックに対してコールドブロックであるので、このようなヘッダの周期的再生は必要である。言い換えれば、ヘッダはフラッシュメモリアレイ401のその他のブロックと比べてサイクリングされる頻度が少ないブロックにある。従って、ヘッダを周期的に再生しなければならないのである。

【0089】ヘッダを再生しなければならない時点を確定するために、ヘッダに隣接するブロックと、ヘッダの消去/プログラムサイクルの数を追跡し続ける。サイクルカウントパラメータは主ヘッダブロックに記憶される。サイクルカウントはヘッダと、ヘッダに隣接するブロックの消去/書き込みサイクルの数を表わす。この別の実施例においては、フラッシュメモリアレイ401の別のブロック、たとえば、ブロック420にヘッダの代替

ク420に記憶されるこの代替ヘッダは、再生動作の実行中に、一時的にヘッダとして使用される。さらに別の実施例では、ヘッダはフラッシュメモリアレイ401の中にはなく、パーソナルコンピュータシステムの一部を成す別個の集積回路カードに含まれるメモリアレイの中にある。ヘッダを記憶するメモリアレイはレジスタ番号アクセス形メモリアレイであると考えられる。従って、ヘッダは別個のシャドウアレイに記憶されることになるであろう。このシャドウアレイを使用する実施例の場合、ヘッダがフラッシュメモリアレイ401の中にないとすれば、ヘッダを再生する必要はないであろう。そこで、ヘッダはコールドブロックにはならず、ホット/コールドサイクリングに関連するサイクルカウントパラメータを含んでいる必要はなくなるであろう。

【0090】ヘッダを使用するか否かは任意であることは明らかであろう。本発明の一実施例では、次に述べる構造によってヘッダの使用を回避している。フラッシュメモリアレイ401は、メーカー及び装置の識別コードを記憶している。一実施例によれば、そのようなメーカー及び装置の識別コードはパーソナルコンピュータシステムのソフトウェアをトリガして、それらの特定のメーカー及び装置識別コードから推定できるヘッダ型情報を記憶しているルックアップテーブルに向かわせる。

【0091】図13は、ファイル構造400の別のクラスタマッピング構成を示す。図13に示す別の構成の場合、クラスタマッピングテーブル446の代わりにセクターマッピングテーブル746を使用する。クラスタセクターマッピングテーブル746をフラッシュセクターハッシュテーブル746ともいう。図13に示す別の実施例の場合、ファイル構造400のクラスタごとに1つのセクターが存在している。さらに、1つのチェーンの中に4つのセクターがある。各チェーンは、順に並ぶ論理アドレスを有する4つのセクターを含む。図13の欄720は、セクターマッピングテーブル746の論理セクターアドレスを示す。たとえば、セクターハッシュテーブル746の記述項751はチェーン0を含む。チェーン0は、4つの論理アドレス0、1、2、3のいずれか1つが検索されたときにアクセスされる。言い換えれば、チェーン0は論理アドレス0、1、2及び3を有するセクターから構成されているのである。セクターマッピングテーブル746の記述項753に見られるチェーン1は、論理アドレス4、5、6及び7を有するセクターから構成されている。テーブル746の記述項761は、全て論理値1であるデータのパターンから構成されるF空(16進数)データパターンを含む。これは、記述項761と関連するチェーンが現在使用されていないことを示す。

【0092】図13に示すように、セクター論理アドレスはアドレスS-1に向かう。Sは区分ごとのセクター数と等しい。セクターハッシュテーブル746中の最後

の記述項はチェーンMである。記述項765はセクターハッシュテーブル746の最終記述項であり、この記述項765はチェーンMに対応する。記述項765はセクター論理アドレスS-4〜S-1と関連している。セクター論理アドレスはアドレス0から始まるので、区分ごとの最終セクター論理アドレスはS-1となる。セクターマッピングテーブル746に記憶される各チェーンは、そのチェーンと関連するセクターに関わる記述項の係リストから構成されている。セクターマッピングテーブル746に記憶される1つのチェーンを構成する各記述項をセクターハッシュ記述項ともいう。

【0093】図13は、セクターハッシュテーブル746のチェーン0に関わる係リストを構成するいくつかの記述項の例を示している。図13に示す記述項は記述項731と、記述項741である。図13に示す記述項731は、セクターマッピングテーブル746に記憶される1つのチェーンの中の1つの記述項を構成するもの示している。記述項731はディスクセクター記述項733と、フラッシュセクター記述項735と、ポイント記述項737とから構成されている。ディスクセクター記述項733は、チェーンを構成するセクターの中の1つの論理アドレスを記憶する。記述項731が係リストチェーン0の最初の記述項であれば、733に記憶されるセクター論理アドレスはセクター0に関わる論理アドレスであるセクター論理アドレス0になるであろう。記述項733に記憶されるディスクセクターの論理アドレスは、パーソナルコンピュータシステムのディスクオペレーティングシステムが見ているアドレスである。図13のフラッシュセクター記述項735に記憶されるアドレスは、記述項733に記憶されているアドレスを有する論理セクターと関連するセクターのフラッシュメモリアレイ401（図8）内部における物理アドレスである。すなわち、記述項735に記憶された物理アドレスを有するセクターは、図8のフラッシュメモリアレイ401の中にある物理セクターなのである。このように、記述項731は1つのセクターの論理アドレスを1つのセクターの物理アドレスにマッピングする、すなわち、相関させる。

【0094】記述項731の物理セクターアドレスが活動優良セクターのアドレスであり、汚れたセクターのアドレスではない場合には、記述項731の記述項737はF空（16進数）データパターンを記憶する。ところが、記述項731の物理セクターアドレスが汚れたセクターのアドレスであれば、記述項731の記述項737は次のセクターハッシュ記述項のアドレスであるポイントを記憶する。図13に示すように、ポイント737はセクターマッピングテーブル746のチェーン0の記述項741を指示している。図13に示す通り、記述項741は記述項741の733で示す場所にセクター論理アドレス2を記憶している。記述項741は、その73

5で示す場所にセクター物理アドレス405をさらに記憶している。アドレス405は、図8に示したフラッシュメモリアレイ401のセクター405の物理アドレスである。従って、記述項741はセクター論理アドレス2をフラッシュメモリアレイ401の物理セクター405の物理アドレス405にマッピングしているのである。チェーン0の係リストの記述項741は737で示す場所にF空データパターンを記憶している。これは、物理セクター405が活動優良セクターであることを表わしている。先に述べた通り、図13に示す実施例の場合、チェーンごとの論理セクターの数は4つである。従って、各チェーンは、チェーンを構成しているセクターごとに1つずつ、少なくとも4つの記述項を有するものと考えられる。セクターハッシュテーブル746は、コード又はデータを記憶するためにフラッシュメモリアレイ401の追加セクターが使用されるにつれて、オンザフライ方式で構成される。コード又はデータを記憶するためにフラッシュメモリアレイ401の追加セクターが使用されるにつれて、新たなチェーンや新たなセクターハッシュ記述項が追加されてゆく。コード又はデータを記憶するために利用できる全てのセクターの割当てが完了するまで、セクターハッシュテーブル746の作成は続く。セクターハッシュ記述項が場所735に物理セクターアドレスを有し且つ場所737にF空データパターンを有する場合、そして、その物理セクターアドレスにより識別される物理セクターが汚れたセクターになった場合、場所737のF空データパターンは重ね書きされて、次のセクターハッシュ記述項のアドレスを指すポイントとなる。その次のセクターハッシュ記述項は新たに追加されたセクターハッシュ記述項であり、活動優良セクターの物理アドレスを含んでいるであろう。

【0095】たとえば、物理セクター405が汚れたセクターになった場合には、チェーン0に新たなセクターハッシュ記述項が追加されるであろう。記述項741の場所737にあるF空データパターンは重ね書きされて、新たなセクターハッシュ記述項を指示するようになる。その新たなセクターハッシュ記述項は、たとえば、論理セクター2を活動優良物理セクター409にマッピングすべきであるならば、場所735にセクター物理アドレス409を含むと共に、場所733にはセクター論理アドレス2を含むものと考えられる。フラッシュメモリアレイ401の各ビットはあらかじめ消去せずに論理値1から論理値0に重ね書き可能であるので、あらかじめ消去しなくともF空データパターンの重ね書きを実行することができる。

【0096】コンピュータシステムは、割当てられた論理セクターごとに、場所737にF空データパターンを含むセクターハッシュ記述項を見出すまで、フラッシュセクターハッシュテーブル746のチェーンを繰り返し走査する。場所737にF空データパターンを含む有効

10

20

30

40

50

セクターハッシュ記述項は、それぞれ、論理セクターアドレス（場所733に記憶されている）を「きれいな」活動物理セクターアドレス（場所735に記憶されている）にマッピングする。これに対し、場所737にF空データパターンが記憶されていない場合には、場所735に記憶された物理アドレスは汚れたセクターのアドレスである。ある別の実施例においては、クラスタごとに2つ（又は3つ以上）のセクターが存在し、また、別の実施例では、クラスタごとに4つのセクターが存在する。そのような実施例の場合、フラッシュセクターハッシュテーブル846は論理セクターアドレスをセクターの物理アドレスにマッピングすることになるであろう。さらに別の実施例では、フラッシュセクターハッシュテーブルは論理クラスタアドレスをセクターの物理アドレスにマッピングする。変更が頻繁に行われるセクターについて長いハッシュチェーンを繰り返し走査する必要性をなくすために、パーソナルコンピュータシステムのRAMに圧縮セクターハッシュテーブルを記憶する。図13に示すRAMセクターハッシュテーブル846は圧縮セクターハッシュテーブルの一例である。RAMセクターハッシュテーブル846を作成するときには、セクターマッピングテーブル746のうち、場所737にF空データパターンを含まないセクターハッシュ記述項を全てRAMには書込まない。場所737にF空データパターンを含むセクターハッシュ記述項のみをRAMに書込むのである。従って、セクターマッピングテーブル746のうち、論理セクターアドレスを活動優良セクターにマッピングするセクターハッシュテーブル記述項のみをRAMに書込むことになる。

【0097】RAMセクターハッシュテーブル846は、セクターの読取り又は書込みが実行されるにつれて、オンザフライ方式で作成される。RAMセクターハッシュテーブル846は性能向上のために使用される。RAMセクターハッシュテーブル846のコピーがRAMから失われても、その事象はフラッシュメモリアレイ401に記憶されているデータの一貫性には全く矛盾を与えない。これは、セクターハッシュテーブル746がまだフラッシュメモリアレイ401に記憶されていると考えられるからである。それにもかかわらず、ファイル構造400の保全性を確保するのを助けるために、本発明の一実施例では、セクターハッシュテーブル746のバックアップコピーをフラッシュメモリアレイ401の空きブロック又は予約ブロックの中にある空きセクターに記憶する。別の実施例においては、論理セクターアドレスを活動優良セクターに直接マッピングするセクターマッピングテーブルをRAM（及びバックアップとしてフラッシュメモリアレイ401の空きセクター）に記憶する。

【0098】図13は、フラッシュセクター割当てビットマップ881をさらに示している。このビットマップ

881の各ビットはフラッシュメモリアレイ401の1つの物理セクターと関連している。たとえば、ビット0は物理セクター403と関連し、ビット1はセクター405と関連する・・・等々である。論理値1であるビットは空きセクターを指示し、論理値0であるビットは汚れたセクターを指示する。従って、ビットマップ881を検査すれば、フラッシュメモリアレイの空きセクターと汚れたセクターを迅速に確定できる。さらに、ビットマップ881の検査によって、連続する空きセクター又は汚れたセクターのいかなるパターンをも容易に確定することができる。

【0099】本発明の一実施例では、パーソナルコンピュータシステムの性能を向上させるために、フラッシュセクター割当てビットマップ881をパーソナルコンピュータシステムのRAMにキャッシュする。図13は、フラッシュメモリアレイ401のブロック状態テーブル444をさらに示している。前述のように、このブロック状態テーブル444には個々のブロックのサイクルカウントを保持する。ブロック状態テーブル444はクリーンアップ動作が終了するたびに更新される。RAMブロック状態テーブル844はブロック状態テーブル444の一次的RAMコピーである。RAMブロック状態テーブル844はフラッシュメモリアレイ401のクリーンアップ中の再割当て動作の間に使用される。

【0100】本発明の一実施例においては、パーソナルコンピュータシステムは、フラッシュディスクエミュレーションファイル構造400と関連して論理状態の反転を実行するためのハードウェア及びファームウェアをさらに含む。論理状態の反転は次のように動作する。従来の固定ディスクやディスクットに関わる従来の標準的なディスクフォーマット動作は、従来の固定ディスク及びディスクットの全てのブロックを論理状態0にクリアするものと規定されている。また、従来の固定ディスク及びフロッピーディスクの物理セクターは、通常、（1）「空き且つ優良」（すなわち、欠陥なし）、（2）「不良」、（3）「予約／未使用」又は（4）「再マッピング」のフォーマットにマーキングされており、標準のデフォルト状態は空き／優良である。従来のフォーマット手順は、通常、固定ディスク及びディスクットのセクター（又はそれより大きい「クラスタ」割当て単位）を空き／優良であるとマークするために「00」コードを指定する。

【0101】これに対し、フラッシュメモリアレイ401のセルは論理状態1にあらかじめ消去される。従って、本発明の一実施例では、フラッシュメモリアレイ401からの出力を反転させるために、パーソナルコンピュータシステムにハードウェアを追加する。FDEシステムハードウェアによってフラッシュ装置出力を反転することにより、フラッシュメモリアレイ401の消去後の状態はシステムには論理値「0」としてあらわれる。

この結果、消去後のフラッシュブロックのデータは従来の当初からフォーマット作成される空き／優良記述子と論理的に矛盾しない。FDEファイルシステムは最大の書込み性能を得るために空き消去済スペースを利用できることを要求するので、消去済スペースをどのように処理するかは重要である。

【0102】本発明の一実施例においては、フラッシュディスクエミュレーション制御ファームウェアは再書込みをできる限り少なくする。このことについて次に説明する。DOSのような従来のオペレーティングシステムはデフォルト時の活動コードとして「00」データを使用するのが普通である。たとえば、DOSの場合、未使用のディレクトリファイル名記述項を「00」によってマークし、次いで、その記述項が論理的に削除されたときに「E5H」データによってフラグ付けする。フラッシュメモリアレイ401に関する上記のような論理状態の反転は、多くの場合に、この第1のファイル名バイトを「00」から有効ファイル名の第1のバイトの有効データへ、次いで「E5H」へと問題なく重ね書きさせると考えられる。論理反転はフラッシュメモリアレイ401の物理データを1の状態から0の状態へ反転させるが、これはブロック全体を別の、あらかじめ消去しておいたブロックに再書込みすることなく実行可能である。一実施例では、FDEシステムの中央ファームウェアに1つのアルゴリズムを追加し、そのアルゴリズムを上述の論理反転ハードウェアと組み合わせて使用する。そのアルゴリズムは、必要な再書込み動作の数を最小限に抑える。アルゴリズムは、まず、どのような論理データ状態の遷移が要求されているかに関してセクター又はクラスタの再書込み動作を検査する。小さなデータセクションを再書込みすべき場合（たとえば、FATの記述項の変更）、その再書込みは物理フラッシュメモリにおける「1」から「0」への遷移によって実行可能であろう。また、その遷移自体は活動有効データ記憶セクター又はクラスタの中の2〜3のバイトを重ね書きすることにより得られるであろう。その結果、クラスタ全体の再書き込みという無駄な動作は回避される。

【0103】本発明の一実施例では、ファイル構造400は誤りの検出と、論理セクター検証とをさらに含んでいる。FDEの実現を最小にとどめることによって、所定の数（偶数）の記憶単位がフラッシュメモリアレイ401の大きな消去ブロック境界と厳密に一致するように、データ記憶単位（セクター又はクラスタ）は規則的な $2^n$ 個の区分（たとえば、512バイトのDOSセクター）に割当てられるであろう。ところが、多くのシステム環境は書込み時には媒体又は伝送の故障を検出するため（そして、おそらくは修正するため）に使用できる2つ又は3つ以上のサイクル冗長性検査（「CRC」）又は誤り修正コード（「ECC」）データフィールドの計算と記憶を要求し、読取り時にはその戻しを要求す

る。

【0104】本発明の一実施例では、ファイル構造400のセクター又はクラスタにCRC又はECCデータフィールドを追加する。この余分のCRC/ECCバイトは、余分のCRC又はECCのバイト数が等しいとすると、最小データ割当て単位の大きさを $2^n + M$ とする。その結果、消去ブロックごとのデータ割当て単位の数は奇数になるか、又は少なくとも $2^n$ の倍数ではなくなる。そのためにFDEシステムファームウェアの複雑さは増すが、媒体の誤りを検出し、さらには修正をも実行する能力を与えることにより、フラッシュメモリサブシステムの総合的信頼性を向上させる助けにはなる。

【0105】記憶単位ごとの余分の「オーバーヘッド」バイトの数を $M+L$ に拡張することにより、クラスタマッピングテーブル446の誤りの検出が可能になる。余分のバイトの数 $L$ は、所定のFDEシステムにより許容されるフォーマット作成可能論理セクターの総数に十分に索引づけしうる数でなければならない。 $L$ に含まれるデータは、その特定の物理フラッシュ割当て単位に割当てられる論理セクター番号である。従って、クラスタマッピングテーブル446がある特定の物理記憶場所を指示している場合、その物理記憶場所が実際にアクセスされたときに論理セクターの割当てを検出することができ。これにより、RAMメモリ装置におけるソフト誤りに起因してRAMイメージクラスタマッピングテーブルで起こりうる誤りを直ちに検出できる。フラッシュメモリアレイ401がソフト誤りを招くことはないので、クラスタマッピングテーブル446のフラッシュベース連係リストバージョンを読取ることによりそれらの誤りを回復できる。余分の論理セクターテーブルバイト $L$ に含まれるデータは、論理セクター／クラスタがその後別の物理セクター／クラスタに再マッピングされるときに重ね書きされるか又は他の何らかの方法によって無効化される必要があるだろう。

【0106】ファイル構造400と共に使用される組み込みフラッシュサイクリング管理は、フラッシュメモリアレイ401の信頼性の向上を助ける。ファイル構造400がバックアップコピー作成の能力を含んでいることも、信頼性の向上を助ける。本発明の実施例は（1）ファームウェアに制御コードを埋め込み且つ（2）クリーンアップ中に直接のブロック間ファイル転送及びディレクトリ転送のために予約ブロックを使用することによりシステムRAM要件をできる限り少なくするのを助ける。ブロック状態テーブル444はフラッシュ装置とブロックの均一なサイクリングを支援し、それは、他の装置及び装置内部のブロック（それらが存在しているとき）に対して消去及び書込みの性能の劣化を発生しかねないホットスポットを最小限に抑える一方で、メモリの信頼性を向上させるのを助ける。ブロック状態テーブル444はさらに予約ブロックを追跡し、それは、クリー

ンナップ及び「バックグラウンドタスク」消去の信頼性を向上させると共に、その速度を増すのを助ける。

【0107】以上の明細書の中では、本発明をその特定の実施例に関連して説明した。しかしながら、特許請求の範囲に記載されるような本発明のより広い趣旨から逸脱せずに、本発明の様々な変形や変更を実施しうることが明白であろう。従って、明細書及び図面は限定的な意味ではなく、例示として考えられるべきである。

【図面の簡単な説明】

【図1】従来のオペレーティングシステムの論理的構成を示す図。

【図2】フラッシュメモリアレイと、コンピュータシステムのRAMに記憶されているファイルシステムドライバとを伴うパーソナルコンピュータシステムを示す図。

【図3】(1)フラッシュメモリアレイと、(2)ROM BIOSソフトウェア及びROM BIOS拡張ソフトウェアに含まれているファイルシステムドライバと、(3)ファイルを転送し且つファイルのディレクトリを更新するためのRAMバッファとを含むパーソナルコンピュータシステムを示す図。

【図4】(1)フラッシュメモリアレイと、(2)ROM BIOSソフトウェア及びROM BIOS拡張ソフトウェアに記憶されているファイルシステムドライバと、(3)別個のシステムコントローラとを含むパーソナルコンピュータシステムを示す図。

10 【図9】フラッシュメモリアレイのクラスタマッピングテーブルの1例を示す図。

【図10】図9のクラスタマッピングテーブル内部の連絡リストの1例を示す図。

【図11】クラスタマッピングテーブルの動作方法の1例を示す図。

【図12】クラスタマッピングテーブルのバックアップコピーの1例を示す図。

【図13】セクターマッピングテーブルの1例を示す図。

20 【符号の説明】

400 ファイル構造

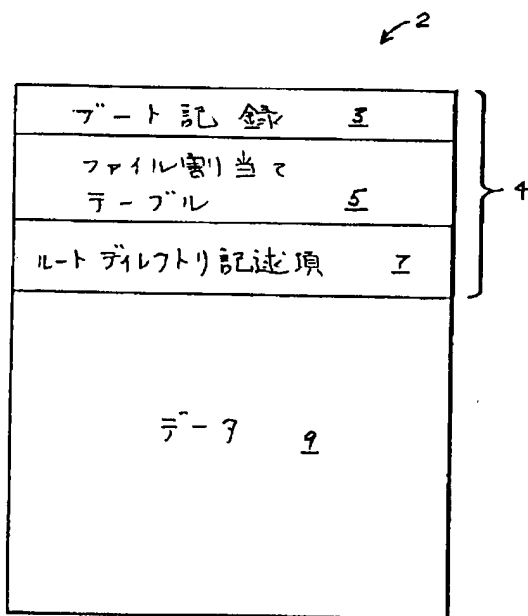
401 フラッシュメモリアレイ

444 フラッシュブロック状態テーブル

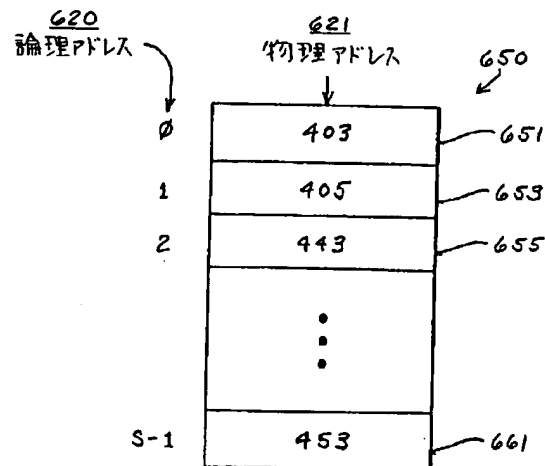
446 フラッシュマッピングテーブル

\*

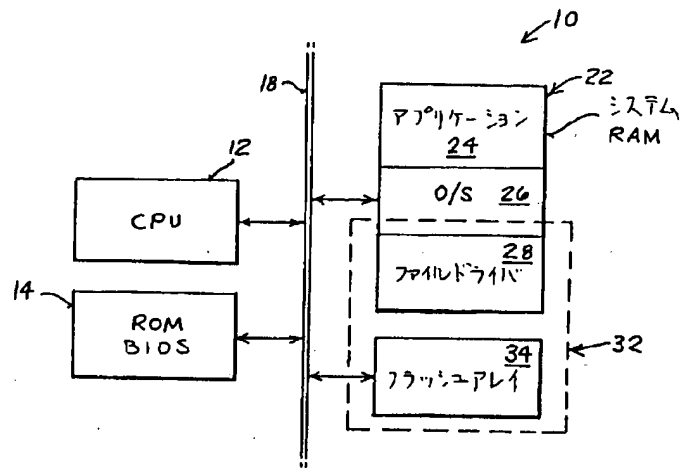
【図1】



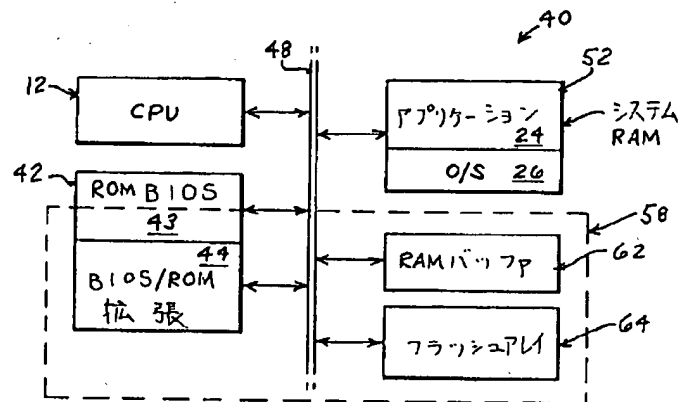
【図12】



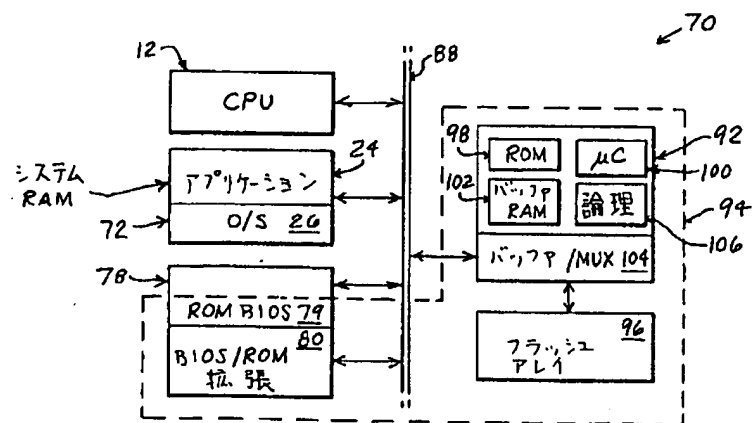
【図2】



【図3】

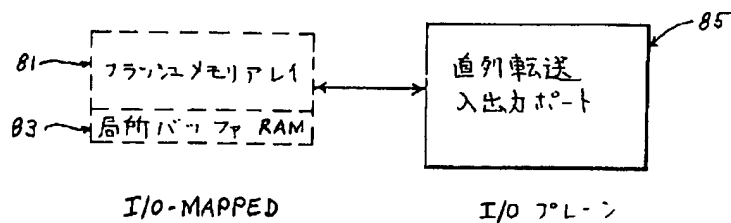


【図4】

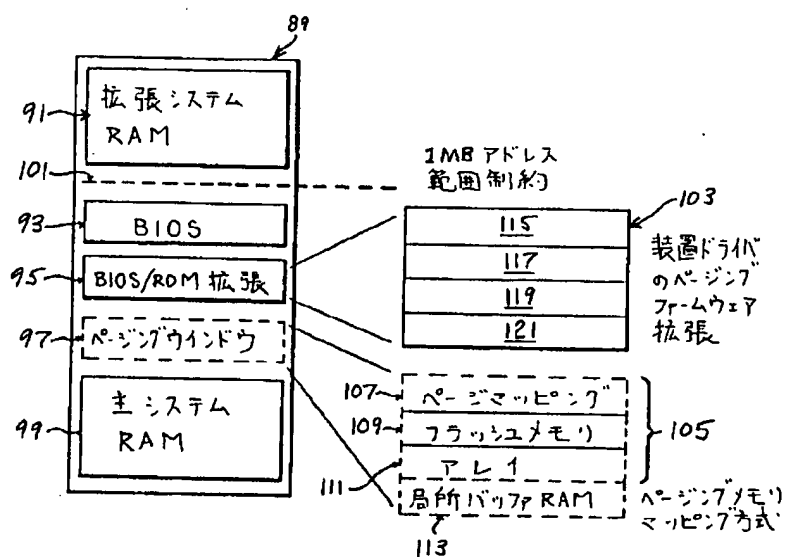




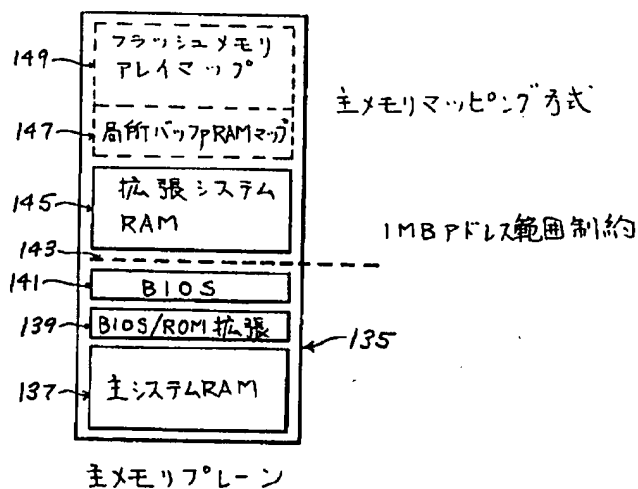
【図5】



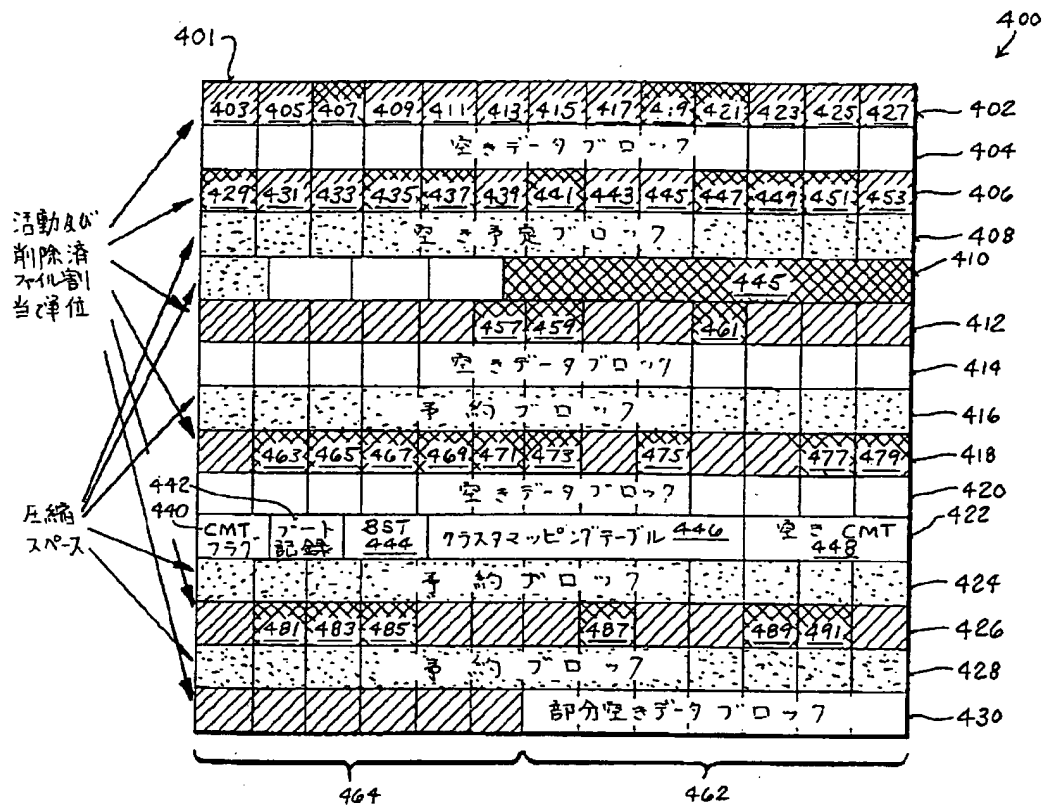
【図6】



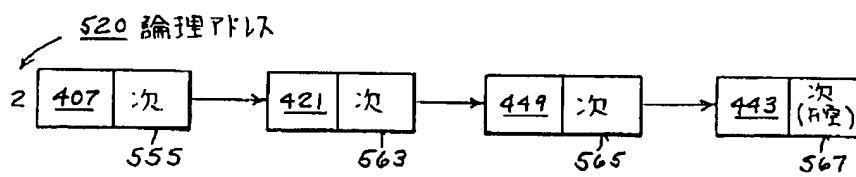
【図7】



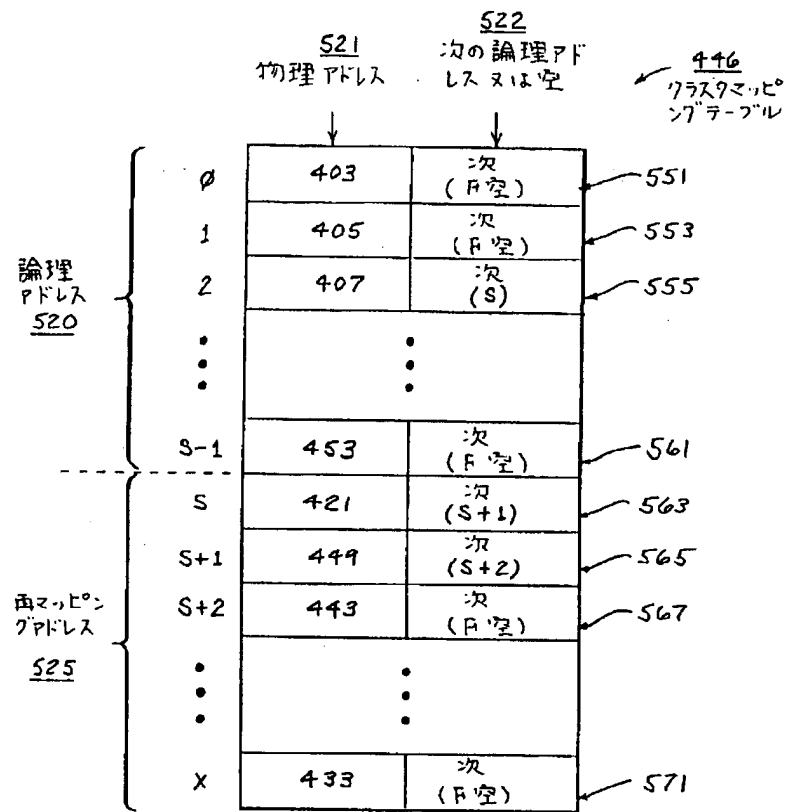
【図8】



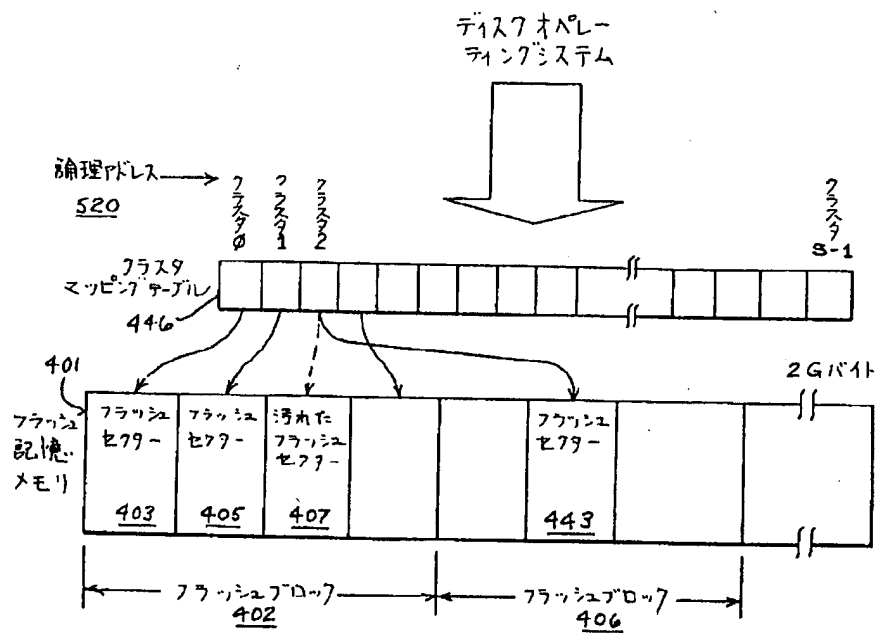
【図10】



【図9】



【図11】



【図13】

